NewTek™

Video Production Systems

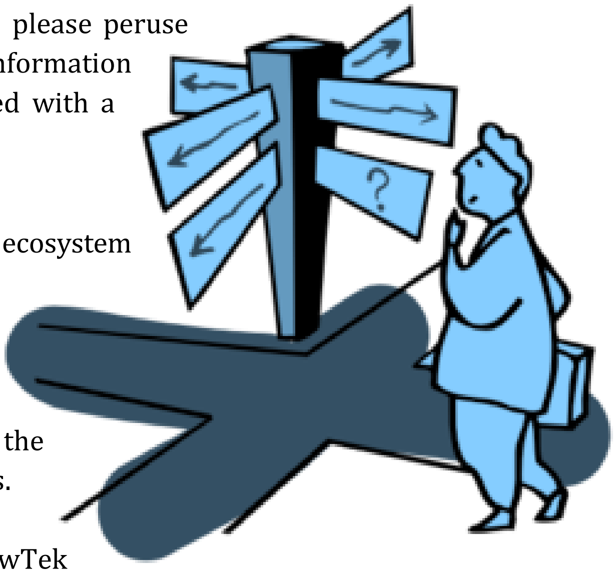
Automation and Integration Guide

Revised – January 20, 2016

CONTENTS

Chapter 1 ABOUT THIS GUIDE

NewTek live production systems deliver awesome production power 'right out of the box'. Their ability to simplify and automate custom operations and workflows, and to leverage the features and content of other platforms in the ecosystem, is the icing on the cake. This guide introduces all of these capabilities.



Even if you are the hands-on, never-ask-directions type, please peruse this page. If any questions arise later, you may find the information here allows you to jump directly to the details you need with a minimum of reading.

- *PART I – OVERVIEW:* The introduction to the NewTek ecosystem also explains the organization of this guide.

  Third-party developers who are interested in participating in the NewTek Developer Program will also find this discussion helpful as an introduction to the way their products can interact with NewTek products.

- *PART II – AUTOMATION:* Introduction to the NewTek product macro system, including the TriCaster™ and 3Play™ macro implementations, all about macro editing and management, and discussion of the numerous ways you can trigger macros. This section also covers mirroring TriCaster 8000 systems.

- *PART III – INTEGRATION:* This section covers cross-product and cross-platform integration and communication.

- *PART IV – APPENDICES:* Information on the NewTek Developers Network, Third Party solutions, a master Macro Command list, and a time-saving comprehensive keyword index.

PART I (OVERVIEW)

This section provides a high level overview of the different components of the NewTek live production ecosystem, and serves a guide to your locating andutilizing those tools that will help you accomplish your production needs and creative goals.

Chapter 2 THE ECOSYSTEM

NewTek live video production systems are everywhere. 'If only they could talk, the stories they would tell'; but wait, they do communicate! And not only with their siblings. More and more, third-party developers have prepared software and systems that can also talk to them, with manifold benefits.

Section 2.1 NO MAN IS AN ISLAND

Appealing as insularity may seem at times, John Donne had it right – "No man is an island". We are inextricably connected.

This is truer now than ever. In the twenty-first century, technology multiplies, extends, and amplifies our connections enormously.

Perhaps that's why, four centuries later, we have the temerity to extend Donne's aphorism as follows: "No *system* is an island". To elaborate just a bit, the more efficient, flexible, and deep the connections between systems are, the greater the rewards in creativity and productivity.

Fluid data transfer between systems is obviously fundamental to modern video production and broadcast. Beyond that, systems with deeply integrated and open communication and control capabilities offer collaborative, efficient workflows and outcomes simply impossible by other means.

Section 2.2 FAMILY AND FRIENDS

At NewTek, we deeply respect these principles. Our systems are engineered with extensive, open and innovative integration in mind. They 'talk' to each other, to make your work more efficient *and* rewarding. In particular, the TriCaster™, 3Play™ and TalkShow™ product families, along with supporting NewTek software and hardware products, enjoy a high level integration.

We also appreciate the importance of 'family friends'. Our customers have diverse needs, and equally unique pipelines. Simple connectivity is important, but much larger workflow benefits can accrue from more advanced interaction. The better we 'play well with others', the more we all gain.

No doubt our obsession with integration accounts for the rapidly expanding collection of collaborative systems and software we refer to as 'the NewTek ecosystem'.

Just as many different members contribute to a literal ecosystem, so too the NewTek ecosystem includes a large and diverse population. The resulting symbiosis provides many, many benefits (sometimes even unexpected ones). The NewTek Developer Network counts among its members many of the most respected veteran contributors to the industry. Alongside these giants you will find many smaller firms offering novel solutions designed to leverage and enhance the already prodigious strengths inherent to the TriCaster and 3Play platforms. And the ecosystem is flourishing, with new members and more innovative solutions appearing at a phenomenal pace.

Section 2.3 Symbiosis and Communication

Communication in one or another form lies at the core of symbiotic relationships. Obviously, NewTek systems can 'listen' for input from external hardware control devices and systems. You'll find options discussed in Section Section 5.1.

Hint: Chapter 6 details a special case wherein one TriCaster controls another, typically to provide redundancy for mission-critical failsafe purposes.

2.3.1 THE REWARDS OF FRIENDSHIP

NewTek live production systems can also exchange a/v streams and metadata, system status details (including 'tally', audio VU levels, etc.), and control instructions with suitably prepared external systems and software.

For example, audio, video, media files, and system control commands can easily be transmitted bi-directionally between systems across a shared network. Third-party solutions can even 'cross-pollinate' with NewTek systems, endowing the latter with dedicated custom macro commands specially designed to work with their product.

The NewTek Developer Program provides members with access to SDKs detailing how to prepare or adapt their products to take advantage of these capabilities for many creative and productive purposes.

Note: Details of the NewTek Developer Program are supplied in in Appendix Appendix A.)

Section 2.4 Taxonomy 101

In general, members of the NewTek ecosystem offer solutions and products that can be classified into the following workflow categories:

- Streaming
- Graphics
- Content
- Automation & Controllers
- Media Asset Management & Storage.

As an introduction to the NewTek ecosystem, a partial listing of third-party products can be found in Appendix B.

Chapter 3 AUTOMATION AND INTEGRATION

This chapter briefly explains various aspects of automation and integration as an aid to understanding the terms and technology discussed in this guide, and provides an overview of how the different elements in these areas work together to offer flexible solutions to satisfy your needs.

Let's briefly consider distinctions between automation and integration as the terms are used in this guide. This is a bit trickier than it might seem, but we want to make the effort as it will enable you to quickly locate the type of information you want without too much tedious searching.

Section 3.1 INTRODUCTION TO AUTOMATION

To begin, let's pay automation its due by giving a nod to its virtually endless benefits. Even in its simplest forms, it can render repetitive operations virtually effortless and error-free at the same time.

The principle *native* engine of automation for NewTek live video production systems is the Macro system. The very same commands exposed for your convenience in this system account for virtually every operation executed by the system.

And, of course, macros can be combined endlessly, with full control over timing. Without much effort at all, you will be able to customize your NewTek system to streamline your workflow and accommodate your personal preferences.

Hint: Chapter 4 provides a thorough introduction to the Macro system.

Without belaboring the point, obviously automation can be extremely simple, or more complex. For example, a simple macro might select a transition and perform an *Auto* to display a specific video source. For convenience, you might assign this macro to a keyboard shortcut.

A slightly more complex macro might load a designated *M/E preset*, select it on the main Switcher's *Preview* row, load a custom transition, perform an *Auto*, and reconfigure the *Audio Mixer* to match the changes. Again, a single keystroke, click, or button press can trigger all of this.

Hint: NewTek live production products support numerous and diverse input methods for triggering macros – see Section 4.6.1 for details.

Let's raise the automation bar even more. Rather than relying on manual input to trigger a sequence of automated tasks, the whole process can be driven by external systems.

For example, an external software application serving as a master control process can execute all manner of complex production operations according to predefined scheduling or other stimuli.

A typical example of this type of automation would include newsroom MOS protocol implementations, which are used to create, edit, manage, schedule and execute virtually every audio, video and graphic element of on air segments in many news centers worldwide.

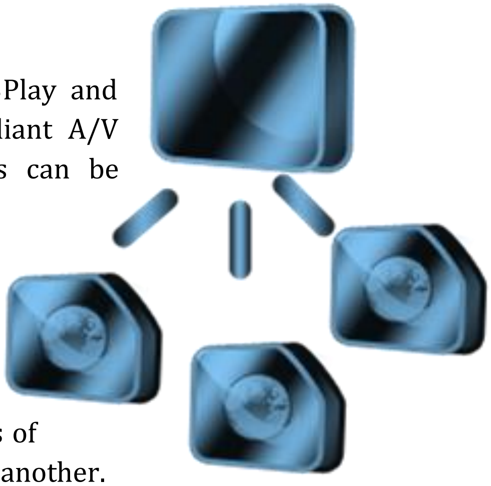
Hint: Some solutions of this type are discussed in Section B.4 of this guide.

By mentioning this level of automation, we have wandered into the rather fuzzy boundary between 'automation' and 'integration'. Before moving on to discuss the latter, let's touch briefly on one rather special automation task.

In high end, mission critical production settings, failsafe systems are de rigueur. One of the more important redundant systems in such settings is the primary video mixer. TriCaster 8000 has unique features that permit it to serve in these environments. Details can be found in Chapter 6, Redundant Control (TriCaster).

Section 3.2 Introduction to Integration

In this guide, we use the term "integration" when referring to broader topics, including cross-platform communication, data transfer and also multi-system control. This may all sound somewhat daunting, but some rather wonderful things are actually readily accessible thanks to 'smart' connectivity inherent in NewTek live production systems.



Let's consider an example. You may already know that both 3Play and TriCaster provide native support for network sources. Compliant A/V signals supplied across standard TCP/IP network connections can be ingested live, just like any other source.

When one NewTek system is linked to another, both are 'aware' of the live connection, and the two systems are able to 'converse' without any further configuration. This allows simple signal traffic, such as tally notification, to pass between systems automatically. More than this, however, the native *Macro* systems of both units allow operators to send instructions from one system to another.

Thus a TriCaster operator could easily create a macro that would i) jump the live video stream from a 3Play networked source back 5 seconds, ii) select a custom "Instant Replay!" transition, iii) commence slow motion playback of the 3Play recording, iv) *Auto* the network source onto *Program* output, v) select a "Back to Live!" transition, and vi) *Auto* back to the original source 15 seconds later – then execute the whole thing with a single click at any time.

Of course, integration with kindred NewTek systems is just the beginning. The chapters in the Integration section of this guide (Part III) also discuss integration with all manner of third party systems and products. This includes, as well, coverage of related topics such as file import and export, and drive formats.

PART II (AUTOMATION)

Full details of the macro system native to the NewTek live production family, along with an explanation of redundant control over TriCaster 8000 systems.

Chapter 4 THE MACRO SYSTEM

Macros can smooth your workflow, reducing complex operations to a single button press, and making it easier to produce sophisticated programs. Macros can also eliminate embarrassing operator errors. Too, the fact that there are nearly endless ways to trigger macros provides many opportunities for both workflow streamlining and creative applications.

Keeping up with the action is one of the hardest things about live production. Fingers can only move so fast, and it can be hard to recall and perform important sequential steps without any slipups.

Macros are the answer to that dilemma. Record a sequence of events and play it back with one click. Or trigger it with a keystroke, control surface, MIDI panel or sequencer, your smart phone, automatically on a *HotSpot* action, on achieving a designated audio threshold, or many other alternatives.

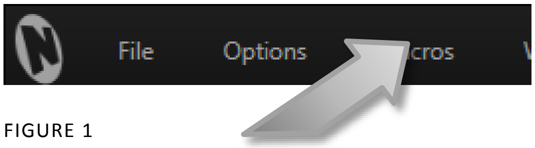
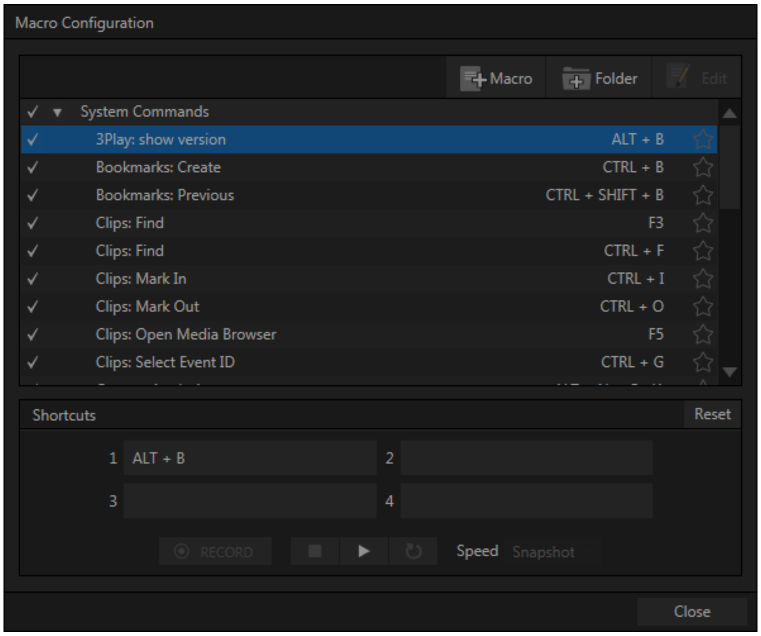


FIGURE 1

Macros can do almost anything. Preload and play content, modify audio settings, automate multi-step sequences or perform synchronous operations. The amazing versatility of macros more than justifies the prominence the *Macros* button gets in the *Dashboard* area of most TriCaster and 3Play models.

Section 4.1 MACRO CONFIGURATION



✓	▾ System Commands	
✓	3Play: show version	ALT + B
✓	Bookmarks: Create	CTRL + B
✓	Bookmarks: Previous	CTRL + SHIFT + B
✓	Clips: Find	F3
✓	Clips: Find	CTRL + F
✓	Clips: Mark In	CTRL + I
✓	Clips: Mark Out	CTRL + O
✓	Clips: Open Media Browser	F5
✓	Clips: Select Event ID	CTRL + G

Shortcuts

1 ALT + B
2
3
4

FIGURE 2

Click *Macros* to show a menu which lists a *Configure Macros* item at the bottom. Select this menu item to open the *Macro Configuration panel* (Figure 2), which in turn enables you to create, edit, and manage your macros.

Note: You may notice some differences in layout of this panel from one product and model to another, but basic functionality is generally as described in this guide.

Section 4.2 System Macros

The largest part of the (resizable) *Macro Configuration* panel consists of the *Macro List*. By default, for any product, this list includes an uppermost entry labeled *System Commands*. Expand this entry by clicking the triangle at left to see a long list of these important macros. The macro entries in this group actually invoke the same shortcuts called by the user interface and *Control Surface* to operate your system (Figure 2 shows typical 3Play *System Command* list content).

Hint: Notice that keystroke shortcuts assigned to macro entries are visible at right.

It's worth noting a few unique aspects of *System Commands*. First, *System Commands* are specially safeguarded within the system. *Rename* and *Delete*, functions normally available from a right-click folder or entry context menu, are disabled.

Hint: If you copy of a System Macro outside that group, the copy becomes editable.

Individual entries in the list can be disabled by un-checking the switch at left; not surprisingly, removing the checkmark beside the *System Macros* folder itself will result in the failure of all 'system default' keystroke shortcuts. By design, this does not affect *Control Surface* operations, however.

Section 4.3 Session Macros

Session Macros is another macro folder that always appears in the list. Macros you create in or move into this special folder are available in the current session (only). This collection gives you a place to collect custom macros that are designed for use within a specific production without cluttering up the list.

Note: The Session Macros group itself cannot be deleted or renamed.

One advantage of the *Session Macros* implementation is that it lets you invoke session specific variants of a macro using the same keystroke shortcut (or MIDI surface button, etc.) without conflicts. For example, you might set up macros that behave similarly in every session, but which point to different content.

Hint: One easy way to copy content from one Session Macros folder to a different session is to Clone the folder and rename it. Then launch the target session, and move the macros you want to transfer from the renamed clone into the current Session Folder.

Section 4.4 Recording Macros

Creating a new macro is simple. Buttons at upper right let you add new folders or macros. Click the Folder button to add a folder, and name it.

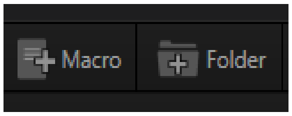


FIGURE 3

Selecting a folder in the list (other than the *System Macros* folder) enables the *Add Macro* button (Figure 3). Click this button to add a new macro entry.

Hint: Double-click directly on the name field for a folder or macro to edit it, or select Rename from the context menu.

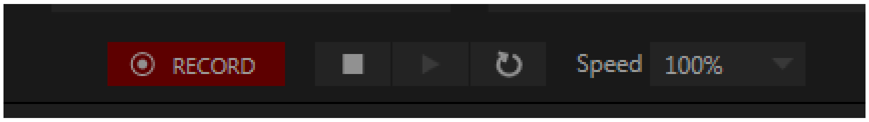


FIGURE 4

Continue to define the macro by clicking the *Record* button at the bottom of the panel, and then just go ahead and perform the sequence of operations you wish to include in the macro. You can use mouse, keyboard, and *Control Surface* operations when doing so. When finished, click the *Stop* button to complete recording.

Test the new macro by clicking the *Play* button (or by double-clicking the macro entry in the list). You'll notice that an animated bar in the background of the macro's entry in the list tracks playback progress. You can set macros to loop using the button at right, or modify the playback rate using the nearby menu.

Hint: You can record a macro that includes other macros. Depending on your order of operations, you may need to re-highlight the newly recorded macro in the list to show its Stop control (to end macro recording).

4.4.1 SNAPSHOT MODE

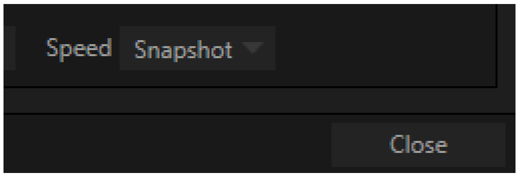


FIGURE 5

One option in the playback rate menu bears explanation: *Snapshot* is rather special. When you choose *Snapshot* as the macro's 'speed', you essentially tell it to jump to its end result. Any operation or delay that is ultimately irrelevant in achieving that end is simply omitted.

Snapshot mode is very useful for macros that configure TriCaster to a particular state. For example, you might want to instantly reconfigure multiple *M/Es* with different angles of a single virtual set for an impending scene change; or perhaps you want to quickly disable *LiveMatte* for all *Media Players* at once. The possibilities are endless.

4.4.2 FAVORITES MENU

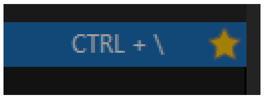


FIGURE 6

You'll see a 'star' gadget (Figure 6) at right for each macro entry in the *Macro Configuration* panel. Click the star to include the macro in the quick access *Favorites* list, shown in the Dashboard *Macro* menu (Figure 7).

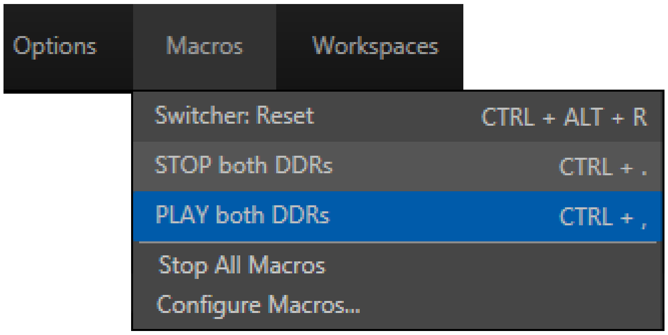


FIGURE 7

Section 4.5 MANAGING MACROS

The *Macro Configuration* panel has numerous features to help you organize and manage your macros, including not only folders, but also rename, clone, copy and paste, and hotkey assignment, as well as *Import* and *Export*.

4.5.1 THE CONTEXT MENU

Entries in the macro lister have a context menu, shown when you right-click an item (Figure 8). Menu items allow you to play or record a macro, delete or rename it. You can, of course, cut, copy, and paste macros, or clone them, combining the latter two operations in one step.

You can also export selections, including multi-selected macros, or even entire folders.

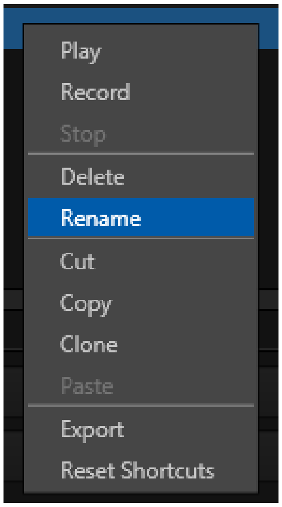


FIGURE 8

The corresponding import item is shown in the menu if you right-clicked either a folder or a blank area in the macro list pane. Import and export can be used to share macros with multiple users and systems, but provide another important service, too.

A good deal of time can be spent preparing complex macros designed to support your production. It would be a shame for these to be lost unintentionally through some mishap, as by some overly-tidy assistant deleting a folder on your day off (or perhaps by performing a *System Restore*). For this reason, we encourage you to use the *Export* feature to prepare a backup archive of your painstakingly designed macros.

Section 4.6 Editing Macros

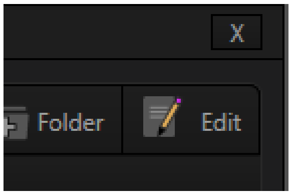
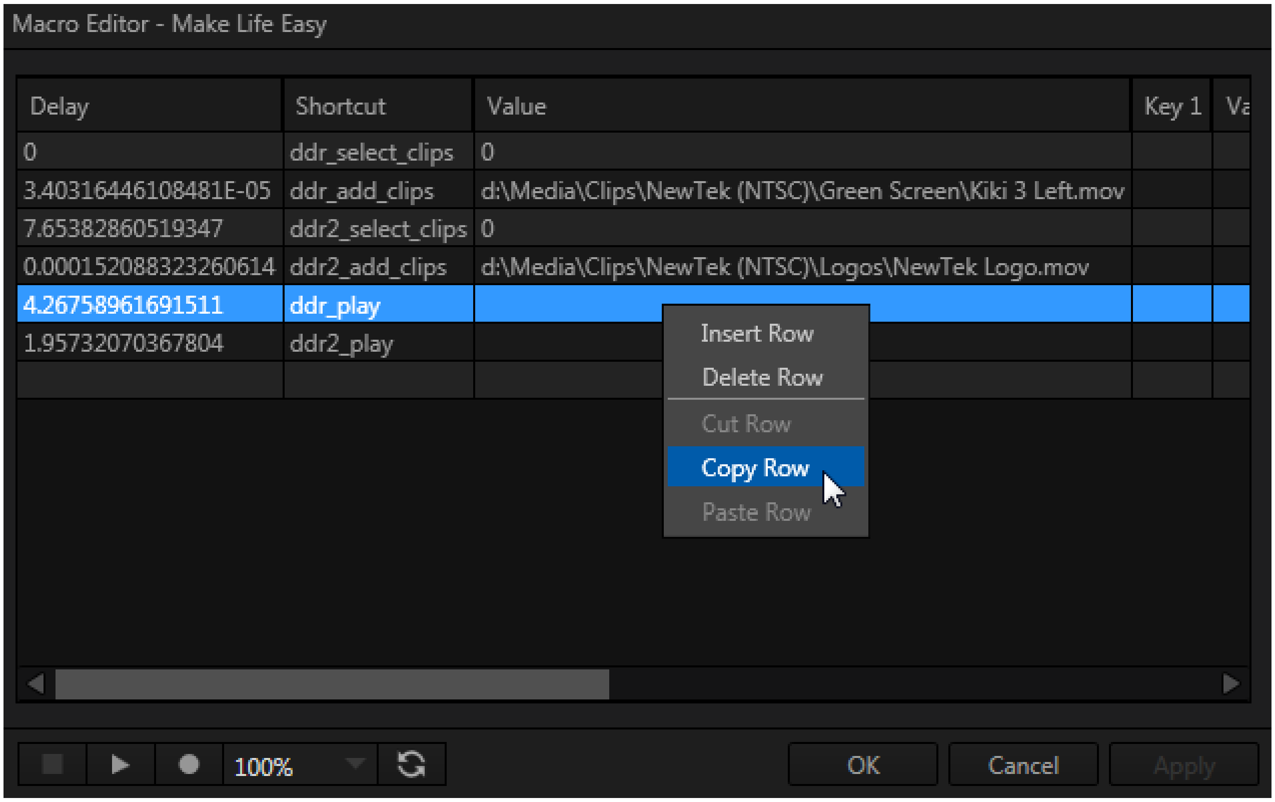


FIGURE 9

Often you will wish to modify values assigned to the various steps in an existing macro, rather than re-recording it; or perhaps you want to experiment with other possibilities. Click the *Edit* button (Figure 9) to open the *Macro Editor* for the currently selected macro.



Macro Editor - Make Life Easy

Delay	Shortcut	Value	Key 1	Va
0	ddr_select_clips	0		
3.40316446108481E-05	ddr_add_clips	d:\Media\Clips\NewTek (NTSC)\Green Screen\Kiki 3 Left.mov		
7.65382860519347	ddr2_select_clips	0		
0.000152088323260614	ddr2_add_clips	d:\Media\Clips\NewTek (NTSC)\Logos\NewTek Logo.mov		
4.26758961691511	ddr_play			
1.95732070367804	ddr2_play			

FIGURE 10

This deceptively simple editor presents the shortcut sequence your macro contains, along with all of its values, including timing information for each line, in a simple to comprehend ‘spreadsheet-style’ interface. Simply click a cell to edit the current entry, or use the arrow keys to navigate.

- Click any cell in the table to select it for editing or other operations. Or click the left-most cell to select a row. Select multiple rows using Shift or Ctrl modifier keys in the usual manner.

- Right-clicking opens the editor’s context menu, which allows you to *Undo*, *Redo*, *Insert* a row (the keyboard shortcut Ctrl + i also inserts a row), *Delete*, or *Cut*, *Copy* and *Paste* selections.

- Standard copy and paste keyboard shortcuts are supported as well. When done editing a macro, click *Apply* (or *Cancel*, to close the editor without saving your changes).

Hint: Use the Record button in the footer of the Editor to directly record new entries to be inserted into the current macro at the selected line

4.6.1 MULTI-STEP MACROS

TriCaster's *Macro Editor* now permits you to create and execute multi-step macros. Adding the line "#waitforcontinue" (or simply, "#pause") to a macro using the *Macro Editor* causes the macro to wait for user input at that step in its execution.

The new *Continue Paused Macro* shortcut, assigned by default to the *backtick* key (`) serves to resume playback. This feature can be used in endless ways, as for example to allow a user to step dynamically through a series of animated CG overlays on demand.

Chapter 5 TRIGGERING MACROS

Macros can reduce or eliminate embarrassing errors. Too, the fact that there are nearly endless ways to trigger macros provides many opportunities for workflow streamlining and creative applications.

As discussed in the previous chapter, one way to execute a macro is directly from the *Macro Configuration* panel, by double-clicking a macro entry, or by clicking the *Play* button (Figure 11).

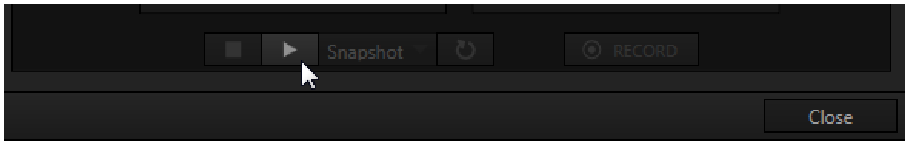


FIGURE 11

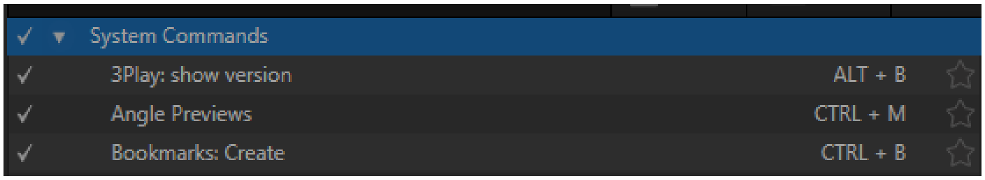
This is just the beginning, however. Macros can be triggered in so many ways that we've devoted a whole chapter to the topic.

For example, macros can be triggered by the following means:

- Keystroke shortcuts
- Control surface buttons
- MIDI pads and sequencers
- GPI signals
- Software events such as:
    - A TriCaster *HotSpot* 'hit'
    - An audio event
    - Or input state change.
- Third-party software applications communicating with your NewTek live production system over a network
- HTTP commands sent by a webpage designed for the purpose

Section 5.1 HARDWARE

As mentioned earlier, macros can be triggered by any of a wide array of supported external hardware devices. Obviously this includes your keyboard; and the majority of NewTek control surfaces compatible with your live production system have a *Macro* button for this purpose.



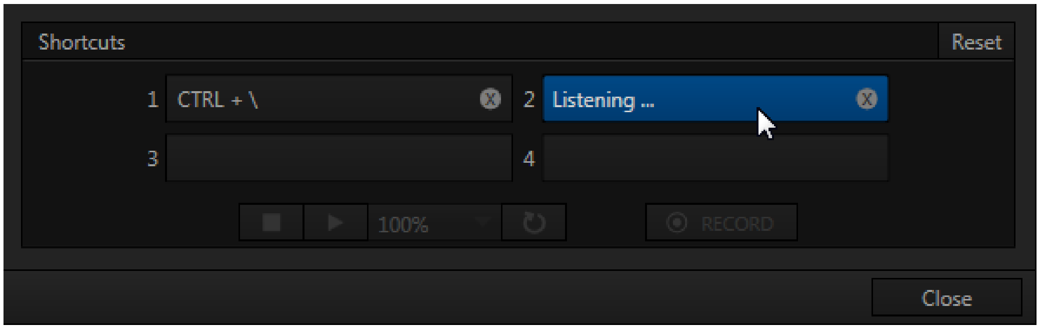
✓ ▾ System Commands		
✓ 3Play: show version	ALT + B	☆
✓ Angle Previews	CTRL + M	☆
✓ Bookmarks: Create	CTRL + B	☆

FIGURE 12

As the simplest example, let's briefly consider keyboard shortcuts before looking into some of the other options.

Of course, many of the *System Commands* entries have default keystroke shortcuts pre-assigned. The first shortcut assigned to a macro (some systems support multiple shortcuts) is displayed at right on the row, near the *Favorites* star mentioned earlier.

5.1.1 KEYBOARD SHORTCUTS



Shortcuts

1 CTRL + \

2 Listening ...

3

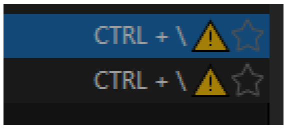
4

FIGURE 13

To set a new shortcut or replace an existing one, click a 'gesture field' in the *Shortcuts* group at the bottom of the *Macro Configuration* panel. It will display a "Listening ..." tag (Figure 13). Then press the desired keystroke.

Hint: For clarity, lower-case characters are uniformly shown as upper-case. True upper-case letters are displayed in the form [Shift + (character)].

FEELING CONFLICTED?



CTRL + \ ⚠ ☆

CTRL + \ ⚠ ☆

FIGURE 14

By the way, assigning identical shortcut combinations to multiple macros *is* supported, and deliberately so. Still, as you may wish to avoid conflicts, a yellow triangular gadget referred to as a 'bang' (or, if you are a 'foodie', a 'nacho') is shown in this case.

Bangs appear at right for all macro entries in the *Macro Configuration* panel lister with shortcut conflicts (Figure 14).

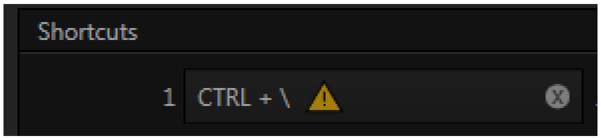


FIGURE 15

Of course, when multiple shortcuts are assigned, the first shortcut for a macro – i.e., the one displayed at right in the *Macro Configuration* panel lister – may not actually be the one that is conflicted; or there can be several conflicts for a single macro.

In such a case, select the macro in the list to show the corresponding *Shortcuts group* entries at the bottom of the panel (Figure 15). Conflicted 'Listen' controls will *all* show bangs. Clicking a bang automatically jumps to the next conflicted entry, so you can advance quickly through the list resolving conflicts as you go.

Obviously, you can resolve a conflict by assigning different keystrokes to conflicted macros. Or you can disable conflicted macros if you prefer, using the checkmark switch.

Hint: Folder level checkmark switches offer a method for managing 'deliberate' shortcut conflicts. For example, the shortcuts assigned to entire folders of macros designed for various purposes can conflict with shortcuts in another folder, but keystrokes for any inactive folders will be ignored.

Deliberate 'Conflicts'

On the other hand, your 'conflicts' may be deliberate; running multiple macros with just one button press or gesture may be just what you had in mind. Pressing the conflicted shortcut key will perform *all* macros sharing that keystroke assignment.

5.1.2 NEWTEK CONTROL SURFACES

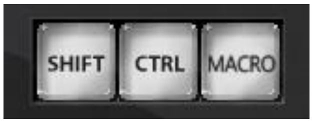


FIGURE 16

Several NewTek manufactured control surfaces feature a dedicated *Macro* button (Figure 16). When this is true, a macro can be assigned to buttons on the control surface in much the same manner as it would be assigned to a keyboard button. You would simply do as follows:

1. On TriCaster's Live Desktop, open the *Macro Configuration* panel.

2. In the macro list, select the macro you wish to assign to a button.

3. Click the mouse in a *Listen* control at the bottom of the panel.

4. Hold down the MACRO button (on the control surface) and press the button you want to assign the macro to.

That's it –close the *Macro Configuration* panel, and test the result.

Hint: When you press the MACRO button, all buttons that currently have assignments light up. This makes it easy to see which buttons are available for your use.

5.1.3 MIDI CONTROLLERS

The MIDI (Musical Instrument Digital Interface) protocol and devices and systems supporting it offers another very useful (and often very affordable) macro trigger option. MIDI devices are often used in the audio and events industries, but can be found in other realms as well. Literally thousands of devices and systems of this sort are available.

The *Macro Configuration* panel system can 'listen' for button presses from most MIDI devices, just as it recognizes input from the keyboard or native control surface.

Note: Many MIDI devices provide 'plug-and-play' convenience. Some, though, require non-standard device drivers. Generally, adding device drivers to NewTek products is discouraged, since these may not have been prepared with the rigorous demands of live production in mind.

If you install a driver and encounter unintended consequences, you can resolve the problem by restoring to factory defaults and, if necessary, updating to the current software version appropriate for your Newtek system.

Too, a wide variety of MIDI software and extensions are available for various platforms, including mobile devices such as tablets and smart phones. These can be used to create unique custom TriCaster control alternatives. See the control surface documentation and addenda for your live production system for more on this topic.

5.1.4 GPI CONTROLLERS

GPI, or General Purpose Interface, is a long-serving analog control signal system based on simple contact closure. GPI inputs and outputs are very common on professional production equipment. The macro system in NewTek live production devices can take advantage of intermediary devices, such as the eBOX™ network/GPI hardware interface from JLCooper Electronics, to support both GPI signal input and output.

CONFIGURATION

For an external GPI device to communicate with a NewTek live production system, it must be manually defined by text entries in the file named gpi_setup.xml. This file can be located in the directory shown below as appropriate for your platform:

- C:\TriCaster\Configuration\
- C:\3Play\Configuration\

The entry for a given GPI control device must contain an IP address and port, password, and custom name, entered as follows:

```
< device  name="name "   ip="###.###.###.###"   port="##"   password=" "/>
```

At the time of writing, the xml 'element name' signified above by the placeholder *device* should be "jlcooper", without the quotation marks. The value for the "name" attribute that follows is a custom name of your choosing.

Hint: Normally, connected GPI devices are identified by unique names in this file; otherwise (if GPI devices share a single name) GPI commands are issued to them simultaneously.

The remaining configuration attributes ("ip", "port" and "password") are set at the external hardware device (refer to the vendor's documentation for details); the corresponding values need only be transferred into the XML configuration file. A typical entry might look like this:

```
<jlcooper   name="JLCooper1"  ip="192.168.128.102"  port="23"  password=""/>
```

Listening for GPI Triggers

Just like keyboard shortcuts, control surface and MIDI button operations described earlier, properly configured and connected GPI devices can trigger macros. To assign a GPI trigger to a macro, simply click a 'gesture field' in the *Shortcuts* group at the bottom of the *Macro Configuration* panel (Figure 13); then send the desired external GPI trigger to the system. The 'listening' control will recorded the GPI signal, and a suitable shortcut entry will be displayed.

Sending GPI Commands

A special macro command allows you to send GPI signals to external devices and systems via network-connected GPI interface devices (such as the eBOX™ from JLCooper Electronics). GPI macro entries are formatted as shown below:

Delay (ms)	Shortcut	Value	Key 1	Value 1	etc.
####	gpi	*name*	*GPI_pin#*	*boolean*	

- **Delay** – the interval, in milliseconds, between the time when the command on the prior line (if any) was issued to the system, and execution of this line.
- **Shortcut** – Use the entry "gpi" in this field to send a GPI signal.
- **Value** – The shortcut value is the name of the GPI device (defined earlier in gpi_setup.xml) that you want the signal defined on this line to address.
- **Key # (0 – *n*)** – The value you enter in this field identifies a target pin on the external DVI device to receive a signal defined in the following field.
  The entry should be formatted as "pin#" (e.g., "pin1", without quotation marks).
- **Value # (0 – *n*)** – This value controls the contact closure state (on or off) for the GPI device pin identified by the preceding key. The value can be entered variously as "1" or "0" **,** "on" or "off", "true" or "false" (without quotations).

A typical entry might look like the following:

Delay (ms)	Shortcut	Value	Key 1	Value 1	etc.
500	gpi	jlcooper	pin12	1	

Hint: Multiple GPI pins can be targetted simultaneously by key/value pairs entered on a single line.

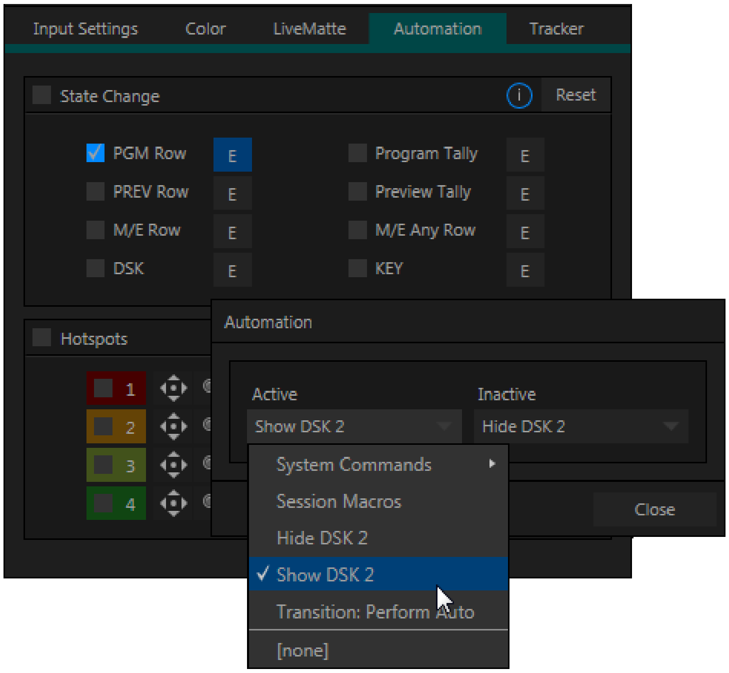
Alternatively, some GPI devices require a GPI 'pulse' of a specified duration. In such a case, you might send an "on" command on one line, followed – after a suitable delay – by an "off" command sent to the same pin.

Section 5.2 Software

As mentioned earlier, macros can also be triggered by software events of various types, including internal or interactive events such as a TriCaster HotSpot 'hit', audio event or input state change, or externally, in response to commands from software applications or even a webpage designed for the purpose.

5.2.1 SWITCHER STATE

The State Change controls located in the *Automation* tab of TriCaster's *Input Configuration* panel allow you to flexibly trigger macros based on the utilization of video sources used in your production.



☑ PGM Row

Active: Show DSK 2

Inactive: Hide DSK 2

FIGURE 17

Macros can now be assigned to run on specific *Switcher* operations, such as:

- *Program* or *Preview* row selection
- Displaying/ hiding the source in a *DSK* or *KEY* channel
- Selecting/de-selecting it on an M/E's A row

- Or any M/E row, or …
- Showing or hiding a source on the *Program* or *Preview* output.

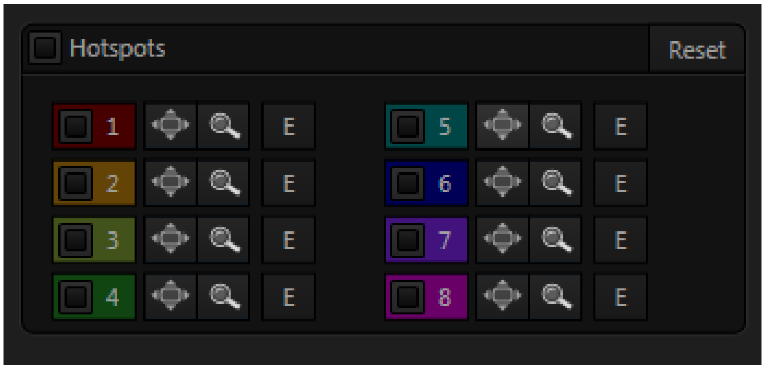
This feature is immensely powerful, and lends itself to all manner of applications, such as the following, to name just a few:

- Automatically fly in a title whenever you switch to a specified remote source
- Then remove it again automatically after it is displayed for a specified time.
- Or automatically select a different *Audio Mixer* preset when you switch from viewing a source in the B monitor of a virtual set on *Program* to displaying it full-screen
- And then change back to the original audio setup when you switch back to the anchor desk.

The possibilities are truly endless.

Simply click the E (Event) button next to a 'state' option for the input and select macros that will run when the source assumes or exits the specified state.

5.2.2 HOTSPOTS



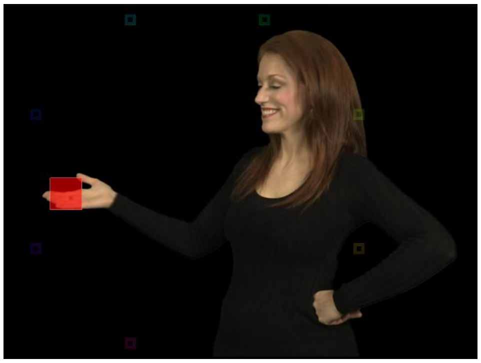


FIGURE 18

FIGURE 19

On-screen *HotSpots,* also configured in TriCaster's Automation tabs, provide yet another way to trigger macros – based this time on activity detected in specially defined regions of the video frame.

Hotspots can serve many purposes. For example, onscreen talent can trigger one macro by moving their hand (for example) into a *Hotspot*, another by moving it out.

- Use live action to play sounds, make *Overlays* and *DSKs* appear auto-magically, or switch the video in a virtual monitor by tapping it with a fingertip.

- Switch from a seated desk shot to a standup virtual set simply set by walking into it; then auto-switch to the next shot when you walk back out of the frame.

- Load up a different *DDR MEM slot*, audio configuration and camera assignments when talent moves from the desk shot to standup in a virtual set.

- Reveal an over-the-shoulder Skype® feed from TalkShow™ and configure audio for the remote interview, then close the picture-in-picture and restore the original sound setup with a wave of the hand

Hotspots are configured in the configuration panels for individual inputs. Double-click the viewport for a camera input to open this dialog, and click the *LiveMatte* tab. The lower portion of this tab contains the *Hotspots* control group.

Note :All Hotspots for an individual source can be enabled or disabled using the switch in the group header, or globally for all sources in a given session using the Options menu.

Hotspots are color-coded, and their respective colors are used to draw the *Hotspot* overlay boxes on your viewports when the *Hotspot Markers* overlay is enabled for a corresponding monitor viewport.

Scale and *Position* buttons allow you to re-size and place the 'trigger zone' for each *Hotspot* accurately.

The *Event* button for each *Hotspot*, marked by a capital "E", opens the *Event Triggers* dialog. This is where you will assign macros that are triggered when something moves into (On Screen) or out of (Off Screen) the otherwise transparent area defined by that *Hotspot*.

FIGURE 20

Hint: Use the Overlay option Flip View Horizontal to let talent see exactly where their marks are on a Multiview screen.

5.2.3 MEDIA PLAYER MACROS

Naturally, *Media Players* get automation support like other *Switcher* inputs, as described above. We didn't stop there, though. *Every* item in a playlist – each clip, still image, audio file or title page – has its very own automation features.

- Any macro you can record or create can be executed automatically on either playback or end of play for any and every individual playlist item.
- Improved multi-selection support in the playlist makes it a breeze to assign macros to multiple items.
- Automatically show titles for certain types of clips and not others.
    - Give them different title page types
    - Use macros to selectively adjust *Proc Amps* on a per-clip basis.
    - Or enable *LiveMatte* keying automatically when needed for certain items.

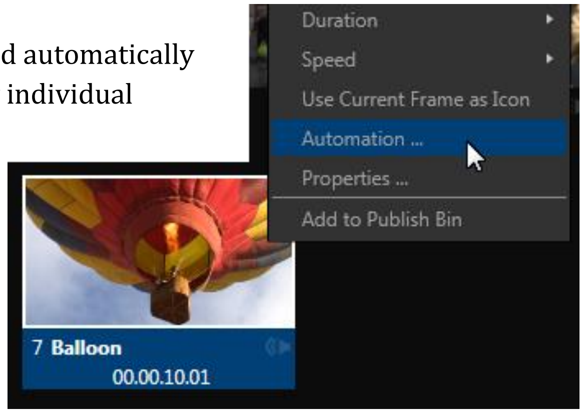


FIGURE 21

5.2.4 AUDIO AUTOMATION

Similar automation functionality is provided by TriCaster's *Audio Mixer*. The related controls are located in the *Automation* group in the *Input Settings* tab of the advanced *Audio Configuration* panel.

Hint: To open this panel, roll the mouse over the control group for any input in TriCaster's Audio Mixer, and click the gear gadget that appears at the top in the label for that input.

The *Automation* control group contains *Follow Program Video* (also known as 'AFV', for Audio Follows Video) and *Macro* options.

FOLLOW PROGRAM VIDEO

Enabling *Follow Program Video* options for an audio source directs TriCaster to track switcher operations affecting the related video source.

Audio for sources with *Follow Program Video* enabled in the *Audio Configuration* panel is automatically removed from mixed outputs until one or more specified video sources are actually displayed on *Program* output.

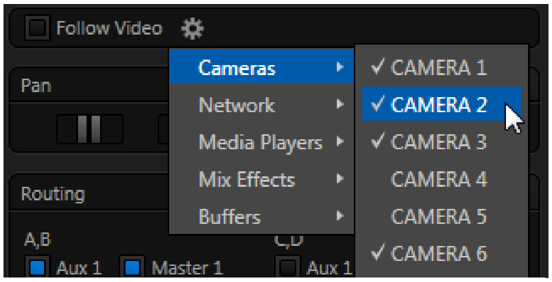


FIGURE 22

RUN MACRO

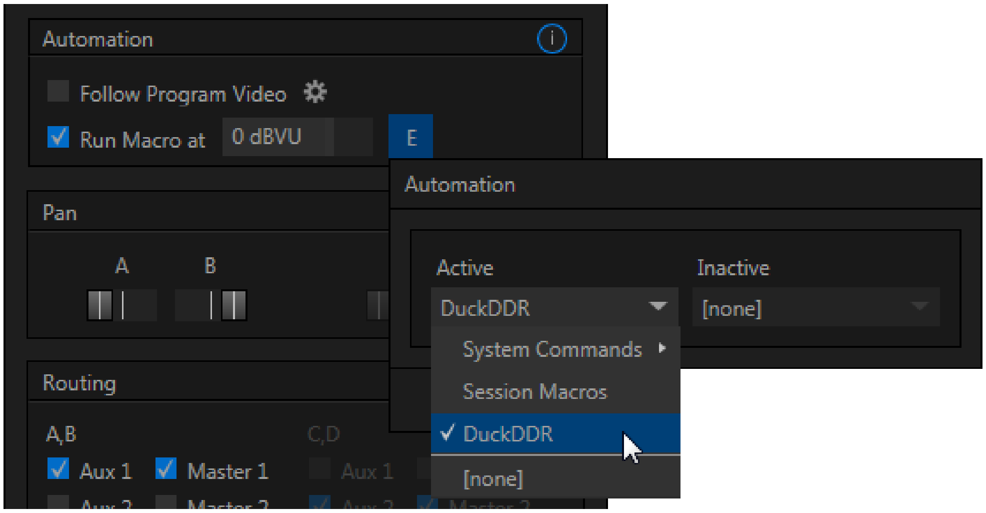


FIGURE 23

Audio threshold triggers allows you to specify a value in decibels to serve as a macro trigger. Whenever the sound level on that input rises above the threshold (or falls below it), designated macros will run

Hint: Transient sounds such as a brief cough are automatically filtered out.

In this manner you could, for example, automatically perform a 'hands-free' camera switch to show someone who begins speaking, and then switch back again when he stops.

5.2.5 NETWORK CONTROL

Means for communicating and controlling NewTek live production systems over a network are discussed in Section 8.3 of PART III (Integration).

Chapter 6 REDUNDANT CONTROL (TRICASTER)

In mission critical broadcast applications and multimedia production environments, it can be vital to provide redundant video mixing architecture. To this end, two or more TriCaster 8000 systems (only) can be configured to operate synchronously. This chapter details how to set up this workflow.

TriCaster 8000's *Options* menu (Live Desktop *Dashboard*) contains the item *Control System(s) Remotely*. Checkmark a TriCaster 8000 listed in this menu.

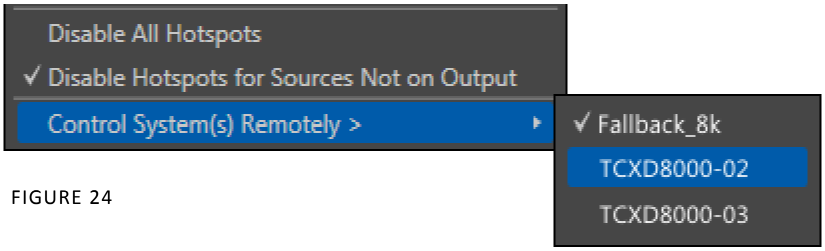

Disable All Hotspots
✓ Disable Hotspots for Sources Not on Output
Control System(s) Remotely >
✓ Fallback_8k
TCXD8000-02
TCXD8000-03

FIGURE 24

Subsequently, all command operations performed at the local unit are echoed to the remote-controlled TriCaster, and it will follow along submissively.

Tip: Mirrored systems should run identical software versions, Rev.1c or better.

6.1.1 'TWINNING' TRICASTERS

For most purposes, media content and all initial control states of both local and controlled systems must be absolutely identical in order for remote control (a.k.a., 'mirroring') to work as expected. Thankfully, achieving this 'twinned' state is not that difficult.

1. Configure the first TriCaster:
    - Create a new session in the desired format.
    - Go on to configure cameras, *Proc Amps*, media content, *Audio Inputs* and *Mixer* settings, *M/E* configurations, etc., just the way you want for your production.
2. Exit the session, and use TriCaster's *Backup Session* feature (see TriCaster documentation) to back it up, gathering all media assets in the process.
3. Click the *Shutdown* icon on the *Home Page*, and select *Administration Mode*.
4. From the *Administration Mode* screen, *Exit to Windows*, and locate the session backup file you created.
5. Transfer the session backup across the network to the remote system.

6. Use the *Restore Session Backup* feature on the *Home page* of the remote system to open the backup session file, and launch the session.

7. Re-launch the original session on the controlling system, and enable remote control over the second system using the *Options* menu item as described earlier.

That does it as far as configuring TriCaster goes. Obviously too, though, mirroring normally calls for upstream distribution amps to multiply camera feeds, attention to matching up audio routing, and so on. Likewise, in most cases, attention must be given to output routing (and sometimes, 'failover' device planning and connection).

In yet another approach to all of this, you might consider using an outboard network drive as the *Session Volume* for both systems.

Note: Normally, TriCasters operating under remote control retain autonomous local control response. You can actually enable bi-directional remote control by configuring two systems to control each other.

This can be especially useful in cases where different operators are responsible for specific aspects of the production process.

PART III (INTEGRATION)

in·te·gra·te [in-ti-greyt] – verb:

1. *to bring together or incorporate (parts) into a whole.*
2. *to make up, combine, or complete to produce a whole or a larger unit.*
3. *to unite or combine.*

Chapter 7 DATALINK™

TriCaster's integrated title page system provides many opportunities for internal automation and broader pipeline integration.

Its unique DataLink™ implementation combines with native as well as third-party automation systems to supply text and image updates for CG purposes from a wide array of live and prepared data sources.

Section 7.1 INTRODUCTION

TriCaster's internal *Media Players* and title system, coupled to a world-class effects engine, allow you to introduce colorful and informative titles and graphics into your productions with ease and flair. And, as discussed in earlier chapters, the macro and automation features add greatly to the value of this implementation:

- The display of title pages can be automatically controlled and timed based on diverse trigger events and Switcher states.
- Using the integrated title editor, text and image content of title pages can be updated on the fly, too.
- Some third-party software solutions can also control and update title page text and image content.

Even so, NewTek's innovative DataLink™ technology extends these capabilities greatly, by providing realtime updates from a wealth of data sources. Let's consider its best known prior role as background.

7.1.1 LIVETEXT™ AND DATALINK™

The optional *standalone* version of LiveText™, NewTek's titling and graphics software, has long derived benefit from its native DataLink support. Title pages prepared and displayed in LiveText can be updated in realtime based on data from a variety of different source types, including files in 'watch folders', data feeds from scoreboards, and more. In turn, the satellite instance of LiveText running on an outboard system can transmit the updated title page display across the local network to TriCaster for use in live programming. (See your LiveText standalone documentation for more detail.)

Section 7.2 TRICASTER AND DATALINK

7.2.1 KEY NAMES AND TITLE PAGES

TriCaster Advanced Edition's *native* DataLink implementation can dynamically update *key name* entries in your title pages. When the page is displayed on output, information drawn from external data sources is substituted for the key name. (The external data is formatted with the attributes you assigned to the key name entry when creating the title page).

You can enter DataLink keys as the source for text or image fields in TriCaster's integrated version of LiveText (provided for page authoring purposes only) when preparing title pages, or later in TriCaster's *live* text editor. Text fields on title pages can contain either literal text or a DataLink key. Let's spend on a few moments considering how you do the latter.

Hint: You can force a minimum number of whole digits before and places after the decimal using the following:
Key: some numeric key
Value: x.x
If we replace x.x with[4.2] and our "some numeric key" is 12.23456789 , then use the following in our Title Page:
%some numeric key[4.2]%
The displayed result will be:
0012.23
Positive and negative values can also be displayed. Percentages can be added like this:
%some numeric key[4.2 percent]%

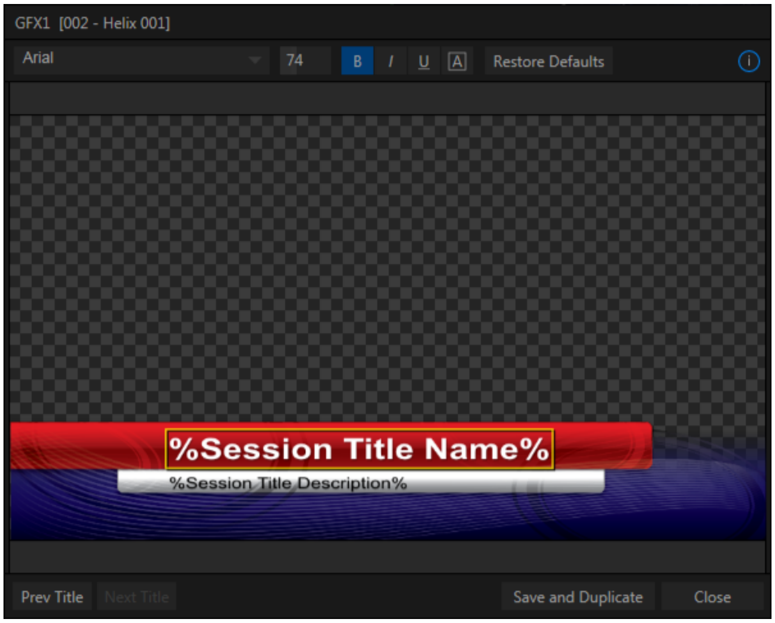


FIGURE 25

In Figure 25, note that the default entries for the name and description lines of the lower-third title page are bracketed by % (percentage) signs.

Hint: See the next section for a discussion of the special %Session xxxxx% keys.

Any text entry surrounded by % signs in this manner is automatically evaluated on display as a DataLink key, and the current value for that key is shown. You can manually enter keys by simply typing them into the field, but there's a method you might like better.

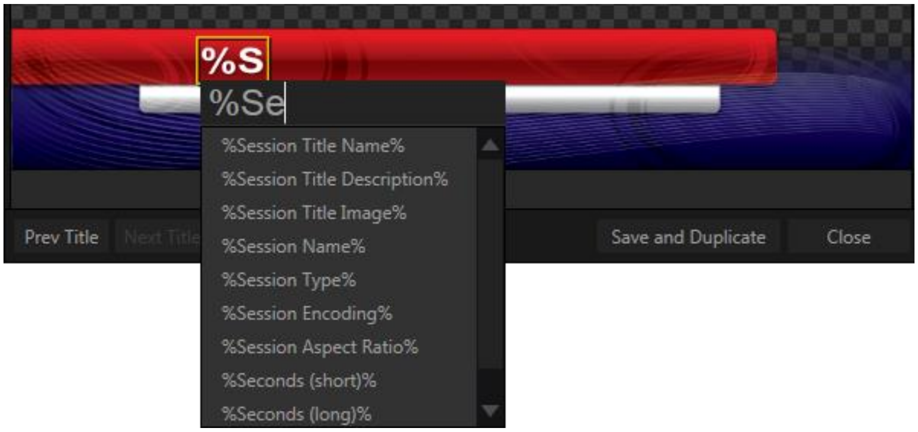


FIGURE 26

If you simply click a text field and type a % sign, all available DataLink keys are shown in a drop down menu. As you continue entering characters, the list updates to show only relevant entries. You can use the arrow keys to highlight the key you want in the menu, and press Enter to select it.

To enter a DataLink key for an image, right click the placeholder image in the live text editor, and select Properties.

Hint: Notice that many internal keys are provided, including keys based on time and day, Media Player metadata (such as %DDR1 Clip Alias% and %DDR1 Clip Comment%), and more. These allow you to easily, for example, use a single title page to automatically show the name and comment for the current DDR clip.

If you then record a macro that displays that page in DSK1 (for example) briefly and then removes it, you can assign that macro to automatically display and hide the correct title for every clip you play from the DDR.

Section 7.3 DataLink™ Sources

While LiveText offer advantages as an outboard CG solution, TriCaster™ Advanced Edition includes an *internal* DataLink™ implementation, which wonderfully complements the integrated title and CG toolset.

TriCaster Advanced Edition's native DataLink implementation extends the original data sources available in several ways. In some cases, support for a given source type has been enhanced; for example, the former ASCII text file support now includes XML and CSV file support. Beyond this, a number of important new internal and external sources have been added.

Here's a list of possible data sources:

- File Watcher
    - ASCII text files
    - XML files
    - CSV (Comma Separated Value) files
- Database
    - MySQL database queries
- RSS (Really Simple Syndication) feeds

- External hardware controllers
    - *Daktronics™
        - Allsport
            - Baseball
            - Basketball
            - Football
            - Hockey
            - Soccer
            - Volleyball
        - Allsport CG
            - Baseball
            - Basketball
            - Football
            - Hockey
            - Soccer
            - Volleyball
    - DSI (Basketball)
    - OES
        - Basketball
        - Hockey
    - Translux Fairplay
        - Basketball
        - Football
    - WhiteWay (Basketball)
    - Whiteway Rainbow (Basketball)
- Internal
    - Time
        - Time
        - Hours (short)
        - Hours (long)
        - Hours (short, 24h)
        - Hours (long, 24h)
        - Minutes (short)
        - Minutes (long)
        - Seconds (short)
        - Seconds (long)
        - AM/PM (short)
        - AM/PM (long)
    - Date
        - Date
        - Day (short, numeric)
        - Day (long, numeric)
        - Day (short)

        - Day (long)
        - Month (short, numeric)
        - Month (long, numeric)
        - Month (short)
        - Month (long)
        - Year (short)
        - Year (full)
        - Time Until Next Event (or end)
    - PGM Source Name
    - PGM Source Comment
    - Session
        - Title Name
        - Title Description
        - Title Image
        - Session Name
        - Session Type
        - Session Encoding
        - Session Aspect Ratio
    - Media Player
        - DDR1 Clip Alias
        - DDR1 Clip Comment
        - Etc.
- Webpage
    - DataLink Web
        - text, including paragraphs
        - images (files or URLs)

* Certain Daktronics controllers (including Allsport 3000 and 5000 models) require an AllSport CG unit to convert the propriety Daktronics feed to serial data to DataLink. Please contact your Daktronics representative for more information.

7.3.1 DATALINK BROWSER EXTENSION AND MORE

As long as the above list might be, it is not complete. In addition, third-party applications can create DataLink keys and supply their values, and you can create and populate DataLink keys using (and within) macros; refer to the information on the “datalink_set” command in Section C.11.

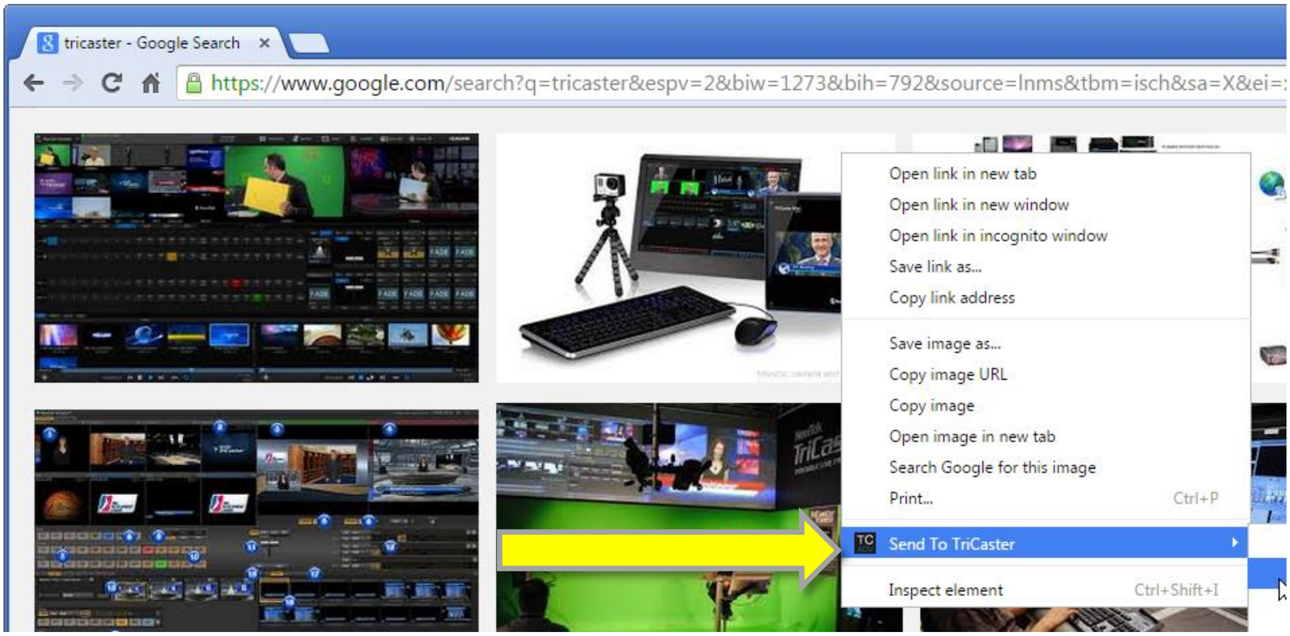

tricaster - Google Search
https://www.google.com/search?q=tricaster&espv=2&biw=1273&bih=792&source=lnms&tbm=isch&sa=X&ei=
Open link in new tab
Open link in new window
Open link in incognito window
Save link as...
Copy link address
Save image as...
Copy image URL
Copy image
Open image in new tab
Search Google for this image
Print... Ctrl+P
TC
Send To TriCaster
Inspect element Ctrl+Shift+I

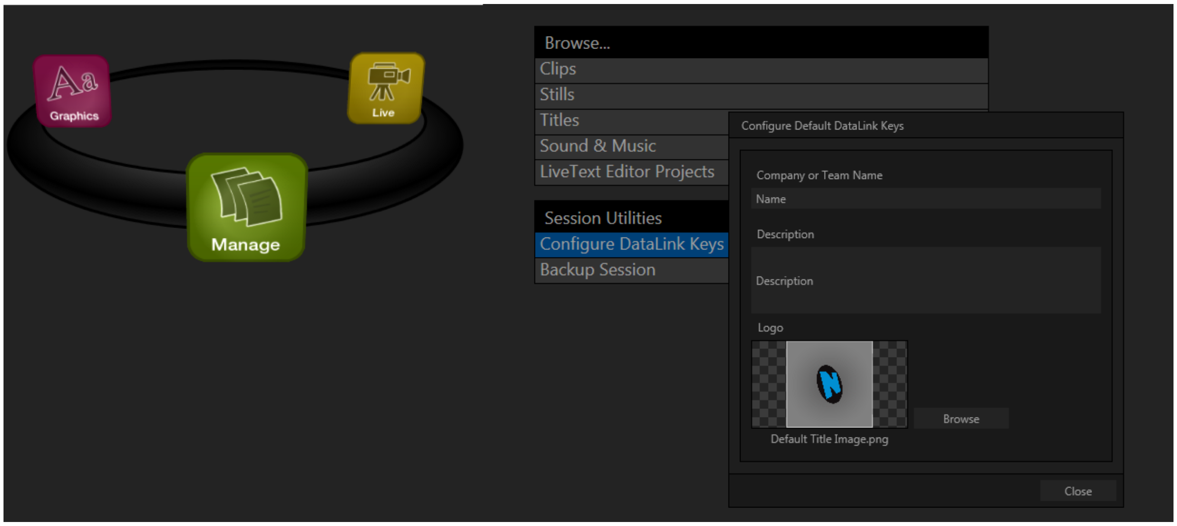
FIGURE 27

One of the most exciting new sources of *DataLink* keys is DataLink™ for TriCaster™, NewTek's new custom extension for the Chrome® web browser. Available without charge from the Chrome Web Store, DataLink™ Web allows you to easily populate both text and image DataLink keys from webpages.

The *DataLink* keys and values are immediately available for use in TriCaster title pages, and elsewhere in TriCaster. Simply select some text, or an image, and use the right-click context menu (or a hotkey) to update a DataLink key you have defined. Any title page using that key will immediately update.

Session Keys

Note that %Session Title Name% and %Session Title Description% are special DataLink keys. Along with %Session Title Image%, the values for these keys are defined in the Startup>Session screen (Figure 28) .



Configure Default DataLink Keys

Company or Team Name: Name

Description: Description

Logo: Default Title Image.png

FIGURE 28

These special ‘session keys’ are pre-assigned by default to appropriate title entries in specific title pages supplied with your TriCaster. When displayed live, these keys are replaced by the values you entered in the Startup screen.

In certain cases, then, this means that simply taking a few moments to choose appropriate values for your company or client will automatically pre-populate stock titles in the session with those values. (Of course, you can modify the title page entries as you see fit, too).

Time and Date

A number of time and date keys are always available in the key insertion drop-down menu. Examples of these key names are Time, Date, and variants on these. These keys allow you to prepare clock and calendar objects that update in realtime on your title pages. When these keys are displayed, the corresponding values are derived from TriCaster’s production clock. This provides many useful and creative possibilities, including counting down to an upcoming events.

7.3.2 FILE WATCHER

Among other sources as described earlier, TriCaster Advanced Edition monitors files in designated *DataLink Watch* folders for changes to keys and their values. The *DataLink Watch* folder system is implemented per-session, allowing you to automatically provide different video programs you produce with unique DataLink key setups. You will find the folder on your local TriCaster host at (*Your session volume*):\Sessions\session-name\DataLink Watch Folder.

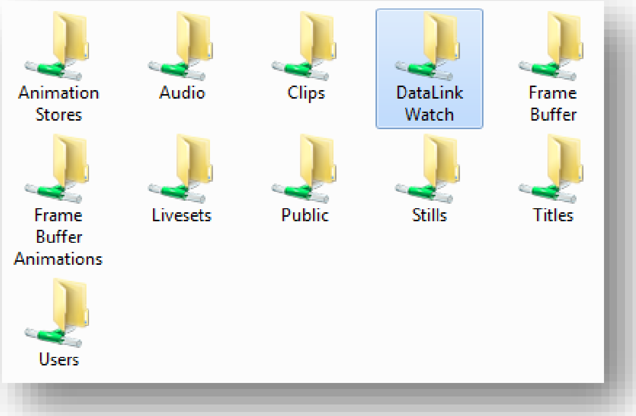


FIGURE 29

Hint: If you enable the Share Media Folders and Buffers option in TriCaster's File menu (Live Desktop), this folder will be accessible to other systems on the network (Figure 29).

ASCII TEXT

DataLink pulls data from ASCII text files (.txt) residing in the (constantly monitored) *DataLink Watch* folder*.* As this is arguably the simplest source available to DataLink, let's use it to demonstrate a few basics before continuing.

1. Create a new text file in the folder (the filename doesn't matter), and open it in a text editor (Notepad will do).

To supply usable values for DataLink, the text files should contain only *key-value pairs*, arranged in the following format: [key] = [value]

Key names from the file(s) will be available as DataLink entries in your title pages. The value you enter beside the key name in the text file will be shown when the page is displayed on output.

Two Key-Value pairs entry examples are shown below:

```
city = San Antonio
temperature = 98°
```

Note: Keys and values may contain punctuation and spaces.

XML

Similarly, XML (eXtensible Markup Language) files can supply DataLink keys and values. Consider the example xml file content provided below:

```
<Key>
 <word1>Hello</word1>
```

```
  <Child>
    <word2>World</word2>
  </Child>
</Key>
```

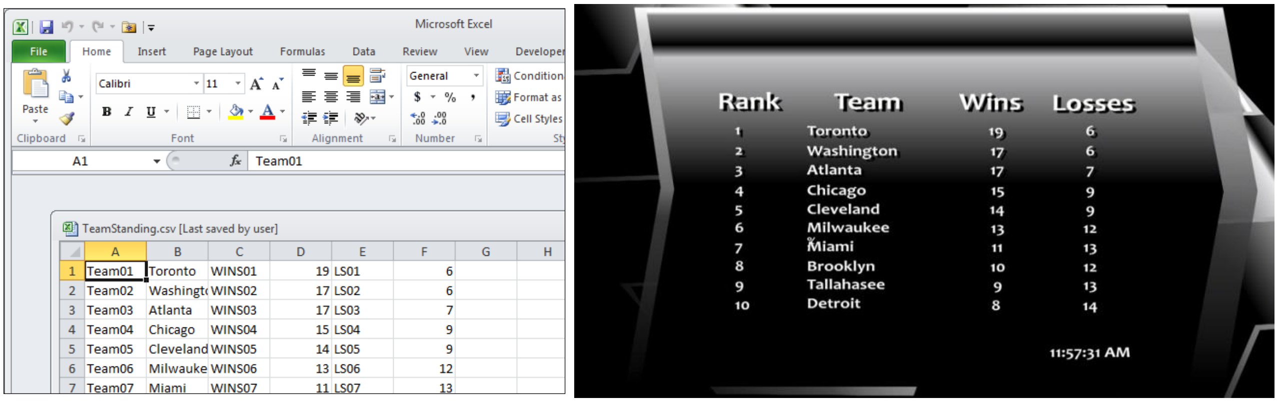
From the entries listed above, the DataLink drop-down will show the following keys with the values listed:

%Key word1% = Hello

%Key Child word2% = World

CSV Files

Imagine using common spreadsheet functions to manage complex sport statistics, then pushing the results to a title page with a single keystroke. That's all possible, thanks to *DataLink's* CSV (Comma Separated Value) file support.



A1 Team01

TeamStanding.csv [Last saved by user]

	A	B	C	D	E	F	G	H
1	Team01	Toronto	WINS01	19	LS01	6		
2	Team02	Washingt	WINS02	17	LS02	6		
3	Team03	Atlanta	WINS03	17	LS03	7		
4	Team04	Chicago	WINS04	15	LS04	9		
5	Team05	Cleveland	WINS05	14	LS05	9		
6	Team06	Milwauke	WINS06	13	LS06	12		
7	Team07	Miami	WINS07	11	LS07	13		

Rank	Team	Wins	Losses
1	Toronto	19	6
2	Washington	17	6
3	Atlanta	17	7
4	Chicago	15	9
5	Cleveland	14	9
6	Milwaukee	13	12
7	Miami	11	13
8	Brooklyn	10	12
9	Tallahasee	9	13
10	Detroit	8	14

11:57:31 AM

FIGURE 30

For example, simply save changes in the CSV file to TriCaster's network-shared *DataLink Watch Folder*, and *DataLink* parses the keys and values it contains, then immediately updates the title page, even if it is on display at the moment.

DataLink parses key-value pairs from neighboring cells on each row as shown in the table below.

Team01	Germany	Team01Wins	6	Team02Losses	2
Team02	Belgium	Team02Wins	4	Team02Losses	1
etc.					

In this case, the key %Team01% would have the value "Germany"; %Team01Wins% would hold "6", etc.

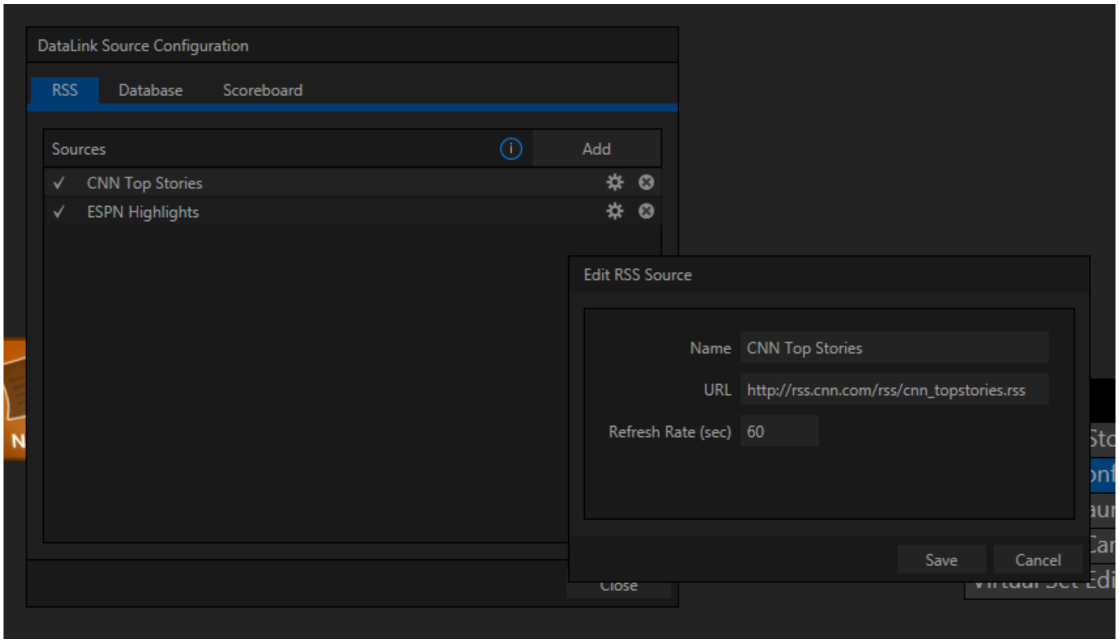
Let's go on to consider the external hardware sources (such as scoreboards) supported by DataLink. TriCaster depends on an external hardware connection to supply values for these keys. In the next section, we'll explain how to connect these external devices.

Section 7.4 DataLink Source Configuration

Internal sources, such as clip comments or time and date keys, do not require any configuration beyond populating them. Some other source types require a little more setup, however. Data from RSS feeds, database queries, and external hardware sources (such as scoreboards) fall into this category.

The necessary settings for these latter sources are conveniently located in the *DataLink Source Configuration* application, launched from the menu shown when you select *the Add-Ons* icon on the ring in the *Startup screen*. The *DataLink Source Configuration* panel has three tabs, *RSS*, *Database*, and *Scoreboard*. The purpose and contents of each is discussed next.

7.4.1 RSS



✓ CNN Top Stories
✓ ESPN Highlights

Name: CNN Top Stories
URL: http://rss.cnn.com/rss/cnn_topstories.rss
Refresh Rate (sec): 60

Figure 31

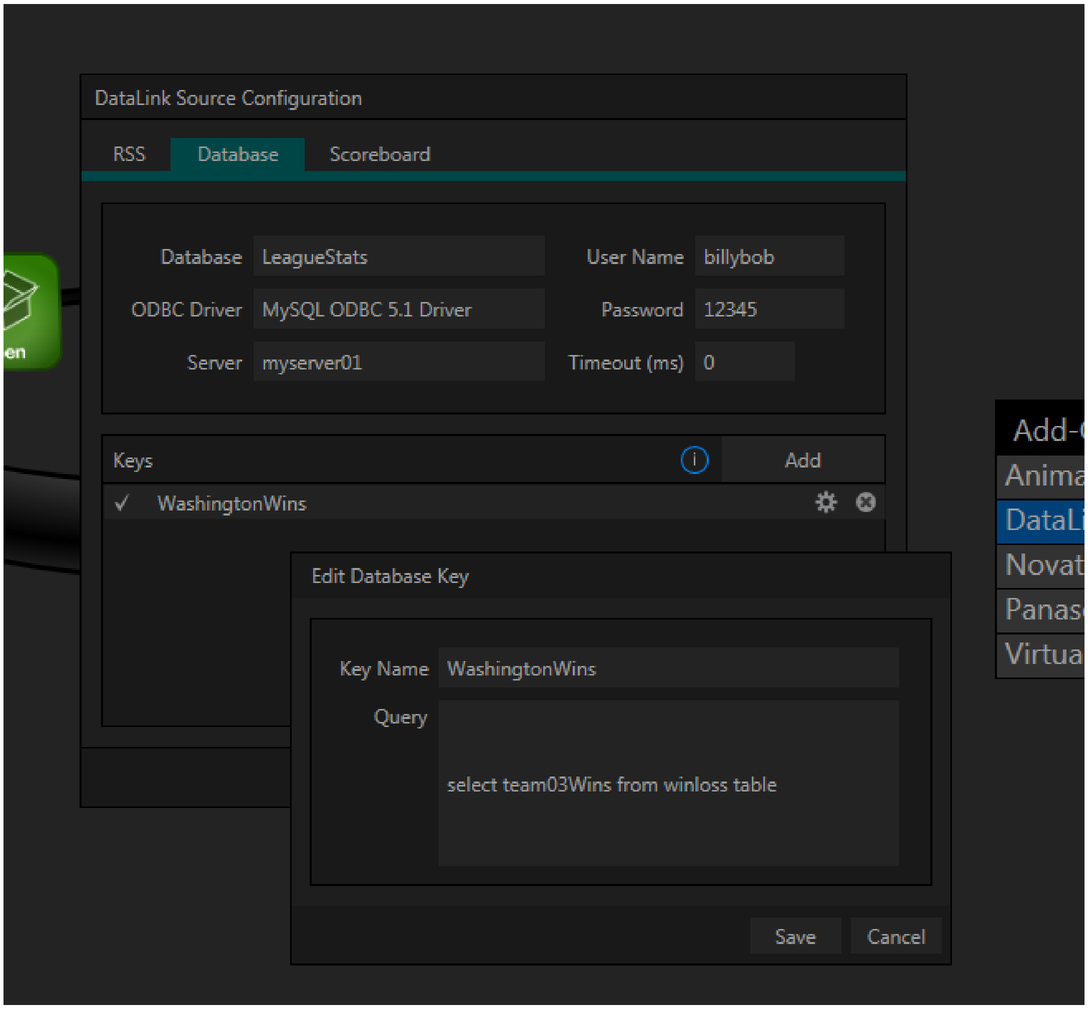
In the RSS tab, click the *Add* button at right to open a dialog that lets you define a new RSS source. Provide a *Name* to identify the new RSS source, and enter the URL to the feed below. The *Refresh Rate* entry below determines how often *DataLink* will poll the source for updates. Click *Save* to store the source (afterward, you can click the gear gadget that appears on rolling the mouse over the source entry to make changes, or the (x) to delete it).

Hint: Key names for RSS feed elements are automatically generated.

7.4.2 DATABASE

For database sources, *DataLink* monitors the value for keys you designate are produced by queries you define. The *Add a Database Key* dialog is shown when you click *Add* in the *Database* tab. Here you can enter a descriptive key *Name*, and the SQL query that will produce the desired value (or values).

Note: You (or someone helping you) will need a measure of familiarity with database addressing and queries.



Database: LeagueStats

User Name: billybob

ODBC Driver: MySQL ODBC 5.1 Driver

Password: 12345

Server: myserver01

Timeout (ms): 0

✓ WashingtonWins

Key Name: WashingtonWins

Query: select team03Wins from winloss table

FIGURE 32

Enter a representative name in the *Database* box (this is simply to help you identify the data source; it need not be an actual file name). Then enter a *User Name* and *Password* for the database in the boxes provided, and specify the driver used for SQL queries in the *ODBC Driver* box. Finally, enter the *Server* name into the corresponding entry box.

Click *Add* to create a new DataLink key. Give it a suitable *Key Name* in the popup panel, and enter the query string that will produce the value(s) you wish to associate with this key into the large box below.

When the SQL query provides more than one match, DataLink creates a key/value pair for each qualified result.

> For example, a keyword "author" could produce an array of matches, which DataLink would arrange as follows:

%author% -> "Voltaire"

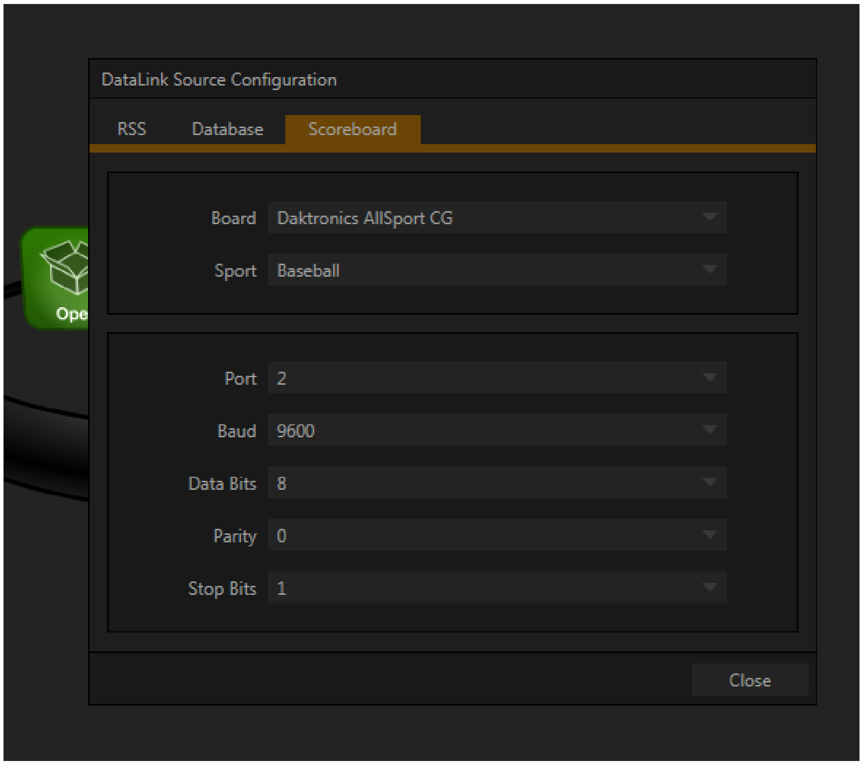
%author.1% -> "James Joyce"

%author.2% -> "Herman Melville"

Click Save to finish the addition of the new key.

7.4.3 SERIAL (SCOREBOARD) SETUP

This DataLink component receives data from compatible external scoreboard hardware controllers. For information on connecting these devices to TriCaster, see Section 7.4.4.



DataLink Source Configuration

Board Daktronics AllSport CG

Sport Baseball

Port 2

Baud 9600

Data Bits 8

Parity 0

Stop Bits 1

FIGURE 33

Once connected, use the *DataLink Source Configuration* utility to notify TriCaster that it is available as a source. Use the *Board* menu to choose the device brand/model you have connected from the list of supported devices. Choose a supported *Sport* in the same manner. The rest of the settings for serial devices auto-fill based on your Board and Sport selections, with one exception - select the Port using the information from the heading

Find the COM Port in Section 7.4.4.

Once you have a supported device successfully connected and configured, the drop-down key insertion menu in LiveText's canvas will list valid key names for that device.

Key Name List

Appendix E, DataLink Hardware Keys lists the actual key available for use with DataLink and the different brands of external equipment it supports.

7.4.4 HARDWARE CONNECTIONS

Note: The steps in this section are mandatory if you require data from an external hardware scoreboard controller.

Naturally, for DataLink to communicate with an external data source, such as a scoreboard, that equipment must be connected to TriCaster and powered up. As well, DataLink must be configured to find and use the connection. We'll discuss how to make and configure connections under this heading.

USB-Serial Adapters

The diversity of supported external systems, cable connectors, and available ports on the host system means this connection may require an adapter.

Newer external devices may use USB connections, but many use older RS-232 (25-pin) connectors (or occasionally, slightly more recent 9-pin) connectors.

Note: Unless the external system is supplied with a USB connection, a USB-Serial adapter is likely required to connect it to TriCaster.

To connect using a USB-Serial adapter, follow these steps:

- Connect the scoreboard controller's output cable connector to the USB-Serial adapter.
  - Connect the adapter to TriCaster.
  - Install drivers for your USB-Serial adapter on TriCaster. Drivers are generally supplied on a Compact Disk (CD) packaged with the adapter by the manufacturer.

    Note that TriCaster may warn you about the dangers of foreign software if it does not recognize the driver for your adapter. (You may wish to ask NewTek Customer Service about supported adapters or request that your favorite be qualified for exemption from these warnings.)

Note: Certain Daktronics controllers (including Allsport 3000 and 5000 models) require an AllSport CG unit to convert the propriety Daktronics feed to serial data for use in LiveText. Please contact your Daktronics representative for more information.

Find the COM Port

The next step involves determining *which* COM port has been assigned to the new connection by the operating system. This information is required to configure DataLink.

- Right-click the *My Computer* icon on the *Windows® Desktop*, and select *Manage* from the menu (to open the Computer Management panel).

- Open the *Device Manager*, and click the + sign next to *Ports (COM and LPT)* in the right-hand pane to disclose available communication ports.

- Locate the entry for your scoreboard controller – take note which COM port number is assigned to it (such as COM 1 or COM2).

> *Note: You should see your new connection listed. If it doesn't appear at first, try removing and re-inserting the USB cable connector – or you can use the "Scan for hardware changes" item in the Device Manager's Action menu. (If it appears, but shows a **!** icon next to its entry, this may indicate a problem with either the USB connection or your driver installation – try re-installing the driver, following the directions supplied with it.)*

- Close the *Device Manager*.

Again, the port number you noted above is required to enable DataLink to recognize the external device.

> *Important Note: In some environments, Windows may arbitrarily reassign the external device to a different COM port following a reboot. If this happens, you could simply update the COM port entry in the affected configuration profile. However, you may prefer instead to lock the connected device to a specific COM port, using the Windows Device Manager.*
>
> *To do this, please locate the current port entry for your scoreboard controller. Right-click the entry name, and select Properties in the drop-down menu. Next, click the Port Settings tab at the top of the Properties panel, and click the button labeled "Advanced". Use the Com Port Number drop-down menu to choose an unused port number, and click the OK button. OK the Properties panel too, then close the Device Manager. The Port Number you assigned should now be retained on subsequent reboots.*

Chapter 8 NETWORK A/V & CONTROL

Most NewTek live production systems support both ingest and output of a/v feeds over standard network infrastructure. This provides a plethora of valuable creative and efficient alternatives. In addition systems can send or receive control instructions from networked devices and systems, offering many powerful possibilities.

Section 8.1 AirSend™

The software API (application programming interface) providing network A/V support in NewTek products is called AirSend™. Various systems and software (both native and third-party) also take advantage of AirSend to send and receive control instructions and tally (on air) notification.

Note: Qualified developers can obtain details through the NewTek Developer Program.

8.1.1 NETWORK VIDEO OUT

NewTek live production systems natively support network a/v signal output (details of network output are provided in the general product manuals.) These program streams can be used by other NewTek live production systems on the network, or supplied to other downstream devices supporting the same video over IP format.

Potential applications of this capability are diverse, and include such things as sending program output to a downstream internet streaming server, a sports telestration application, etc. The downstream system need only be able to display the MPEG-2 stream received.

For example, the popular open source media player VLC can play the network output stream from a NewTek live production system. It is simply necessary to open a URL formatted as shown below in the VLC application:

http://ip_number:port_number/NewTekNetworkSend.mpg

- Substitute the IP number of the source system for "ip_number" above.
- The port number for network outputs from TriCaster defaults to 49480. 3Play's A and B outputs default to ports 49480 and 49481. (If the default ports are unavailable, the system attempts to access the next higher port, and so.)

8.1.2 NETWORK VIDEO IN

Likewise, video (complete with embedded alpha channel and audio) served to NewTek production systems across standard network architecture can serve as live a/v sources. NewTek systems support this workflow by means of 'Network Inputs'. TriCaster Pro models sport two such sources. 3Play 440 and

3Play 4800 provide access to a network feed via their DSK source menus. (Full details can be found in the User Guide for your NewTek product.)

Qualifying sources are listed in the source selection menus for network inputs automatically. When selected, these sources also gain immediate access to AirSend's network communication features, without any further need for configuration.

The following are just some typical network input sources:

- Video program output from other NewTek live production systems
- CG sources from applications such as NewTek's own LiveText™ or third party CG solutions (including many products from leading broadcast industry vendors)
- Live screen captures from connected computers, webcams sent by NewTek's included iVGA™ application, and more.

Let's discuss a few of these briefly.

EXAMPLE SOURCES

TriCaster and 3Play

As just mentioned (Section 8.1.1), NewTek live production systems can send video over IP to downstream systems on the same network (specifically, systems should be on the same sub-net). More detail can be found in the User Guide for your NewTek product.

iVGA™

iVGA™ is the proprietary NewTek utility supplied to let you use video directly from the interface of a networked computer. Source video, and in some cases, audio, is sent directly to a Network input.

iVGA has a tiny footprint, and can even be run from a USB thumb drive, without installing it directly on the remote client system. Clients are supplied for both Microsoft Windows® and Apple OS X®. iVGA installation files are supplied in C:\TriCaster\Extras\iVGA or C:\3Play\Extras\iVGA on your NewTek system. Full documentation can be found in the User Guide.

AirPlay®

AirPlay®, is an Apple® protocol for sending audio and video between an AirPlay source (which may be a computer, or a mobile device such as an iPad® or iPhone®) and another device across the local network. NewTek system's Network Inputs provide native support for AirPlay sources (note, AirPlay 'mirroring' is not supported). Full details can be found in the User Guide for your NewTek live production system.

LiveText™

NewTek's own LiveText™ 2 gives you the flexibility to add a dedicated title station to your live production. Build titles and graphics on any laptop or computer and send them directly to the network inputs of NewTek live production systems.

Make your production pop with network-style scrolls, crawls, title pages and lower thirds. And use the integrated DataLink™ application to display real-time data with instant updates: connect directly to a number of popular game board systems to access scores, times, etc., or access other supported realtime data sources, including RSS, SQL queries, and so on.

For further information, please visit:

http://www.newtek.com/products/tricaster-software/tricaster-livetext.html

8.1.3 AIRSEND CONTROL

AirSend supports more than just a/v data transfer. For example, one form of information available to network sources connected is tally (on-air) notification.

Beyond this, though, AirSend also provides the ability to transmit commands between NewTek live production systems and other connected sources. This provides extremely powerful capabilities.

For example, IP-based Viz Engine™ and Viz Trio™ graphics systems from Vizrt® are able to select and display CG pages, perform animations and more based on instructions from the macro system native to NewTek's TriCaster and 3Play systems. This permits you to operations between NewTek and Vizrt products for sophisticated live production workflows with centralized control.

Note: AirSend is just one way for external systems and software to communicate with and control NewTek live production systems. For example, control applications can also be prepared using simple TCP/IP commands. The NewTek Developer Network program provides members with SDKs detailing the various communication methods available.

EXAMPLE – CONTROLLING 3PLAY

Let's consider a simple example taking advantage of the potential this ability offers. Using TriCaster's Macro system (see Chapter 4, The Macro System), it is easy to take advantage of AirSend to send instructions from TriCaster to 3Play via the Macro system.

Such a macro could actually perform sequential operations on both systems, and be assigned to a shortcut keystroke or other macro trigger (see 4.6.1, Multi-Step Macros) for convenient execution with a single user interaction.

When both 'parties' to the network 'conversation' have been prepared using the AirSend API, as in this case, there is no need for complicated configuration. TriCaster 'knows' which network source is connected to its network inputs, and automatically creates the necessary bi-directional communication channel.

For example, 3Play network outputs typically appear in TriCaster's Network Input source selectors as something like "3Play(A)" and "3Play(B)". If you select "3PlayA" as the active source for TriCaster's "Net 1" input, a communication channel between 3Play's A channel and Net 1 is automatically opened. Unique shortcut entries in the macro identify which Network Input the instructions specified will be directed to.

When examined in TriCaster's Macro Editor, a typical macro entry of this type might look as follows:

Delay (ms)	Shortcut	Value	Key 1	Value 1
0	net1	clip_play	speed	.5

The shortcut "net1" defines which Network Input will be addressed.

Additional entries on the same line constitute an instruction that will be performed when the macro is executed.

- The Value "clip_play" commands the target source to play the currently selected clip (in our example, the current clip selected on 3Play's channel A)

- The Key 1 entry "speed" sets the playback speed to the value which follows next on the line, ".5" in our example. This will cause playback to occur at 50% speed.

It is entirely possible to create more complex macros that combine AirSend commands of the sort just mentioned

Default AirSend Commands

As mentioned earlier, the macro shortcut entry "net1" addresses a controllable network source connected to the first network input on a NewTek live production system. TriCaster has a second network input, which you can send commands to using the shortcut entry "net2".

The AirSend API also allows third-party developers to implement custom commands suited to their requirements. These, when provided, can be used in macros just like 'NewTek native' commands, even benefiting from the same automatic communication channel configuration. Documentation supplied with third-party products will provide information on custom commands that have been included.

By default, third-party products generally support the AirSend commands discussed next, included in the NewTek product macro system.

Clip_Store

Delay (ms)	Shortcut	Value	Key 1	Value 1
(ms)	*net1 or net2*	clip_store	index	*ID*

This command stores a custom local reference ID for the current clip (the one currently visible at on the network input). The value *ID* can be a string. ID is global and shared across your system (it is not stored per system output, if you have multiple).

For example:

Delay (ms)	Shortcut	Value	Key 1	Value 1
.0001	net1	clip_store	index	AAA1

The entry above will ‘remember’ the current clip with the name “AAA1”. (The default for ID is an empty string, which is a valid storage target.)

Clip_Restore

Delay (ms)	Shortcut	Value	Key 1	Value 1
(ms)	*net1 or net2*	clip_restore	index	*ID*

This command cues up content previously stored with a specified ID value on the upstream source channel assigned to the network input designated.

For example:

Delay (ms)	Shortcut	Value	Key 1	Value 1
.0001	net1	clip_restore	index	AAA1

The clip previously indexed as “AAA1” (using clip_store) is restored on the source system output channel connected to Net 1. The playhead is set to the beginning of the clip. (If the indexed clip is not located, nothing occurs.)

Clip_ Select

Delay (ms)	Shortcut	Value	Key 1	Value 1
(ms)	*net1 or net2*	clip_select	index	*#*

Select a page (or clip) defined by the value assigned to index. This may be a number specifying a particular page or at times, another property.

For example, sending a “clip_select” command to 3Play with a suitable numeric value assigned as the “index” key selects a specific Play List tab by index (assuming Play List mode is active. On the other hand, in Clip List mode, if the value for “index” was “0-023” the clip referred to would be selected.

For example:

Delay (ms)	Shortcut	Value	Key 1	Value 1
.0001	net1	clip_select	index	4

This would select the fourth Play List tab on the 3Play output (A or B) connected to Net 1.

CLIP_ MOVE

Delay (ms)	Shortcut	Value	Key 1	Value 1
(ms)	*net1 <u>or</u> net2*	clip_ move	distance	#

Move the specified number of pages forwards or backwards from the current page.

For example:

Delay (ms)	Shortcut	Value	Key 1	Value 1
.0001	net1	clip_move	distance	-1

The entry above would select the previous clip on the source connected to Net 1.

CLIP_ PLAY

Delay (ms)	Shortcut	Value	Key 1	Value 1	Key 2	Value 2
(ms)	*net1 <u>or</u> net2*	clip_ play	speed	#	position	#

You can specify "speed", "position" or both keys (the order of keys is not important).

When "position" is not specified, play begins at the current frame. The value for position is specified in seconds, while speed is expressed as a playback rate value (1.0 = 100%).

For example:

Delay	Shortcut	Value	Key 1	Value 1	Key 2	Value 2
.0001	*net1*	clip_ play	speed	-.5	position	10

This entry would play a clip backwards at 50% speed from a position 10 seconds into the clip.

CLIP_ SCRUB

Delay (ms)	Shortcut	Value	Key 1	Value 1
(ms)	*net1 <u>or</u> net2*	clip_ scrub	distance	#

This command will move the playhead backward or forward by a distance of # seconds.

For example:

Delay (ms)	Shortcut	Value	Key 1	Value 1
.0001	net1	clip_scrub	distance	5

The entry above would advance the playhead five seconds further into a clip displayed on the source connected to Net 1.

NOTES: The software associates values with their key name, thus key can be entered in any order. The following are valid formats for numeric entries: "+0.1", "0.1", ".1", "-0.1".

Section 8.2 AMP

3Play™ models 440 and 4800 provide support for another control protocol called AMP (Advanced Media Protocol). AMP commands are quite similar to the AirSend commands just discussed, and provide another means by which compatible systems can issue commands to 3Play in support of sophisticated workflows.

For example, Carbonite™ video switchers from Ross Video® support AMP over an Ethernet connection. The switcher panel can control NewTek 3Play instant replay systems, treating them as video servers and accessing and clips, playlists, or delayed video streams. The AMP integration provides full play-list and queuing support, clip query and more.

NOTES: A document named "AMP Video Server Setup", available from the Ross Video® website, details how to configure a Carbonite system for AMP communication.

Section 8.3 TCP/IP

It is also possible to communicate with TriCaster and 3Play over a standard network TCP/IP connection. Custom applications running on a networked host system can directly control the functionality of most NewTek live production systems by this means. The methods involved, including how to connect to and control NewTek systems, are detailed in documents available to members of NewTek's Developer Network.

Section 8.4 HTTP

For TriCaster Advanced Edition and 3Play Mini (only), TCP/IP communication is augmented by HTTP methods that are easily accessed by advanced end users. Using a simple text editor, someone conversant with HTML can easily create useful 'applets' run in almost any web browser that can interoperate with supporting NewTek systems. Add programming knowledge (say, javascript or Python for example) and a whole world of possibilities open up.

Most current NewTek products also act as HTTP servers, and are able to receive either GET or POST messages. The internal web server can be addressed as in this example:

http://*TriCasterSystemName*

Hint: In order to determine if the session is running, you can query: http://*TriCasterSystemName* /v1/live
This will return a text string of TRUE when in a session, and FALSE when in the control panel.

8.4.1 GET COMMANDS

You can send shortcut commands and (macro) trigger messages by using http GET commands. To send a shortcut, you would simply get the address:

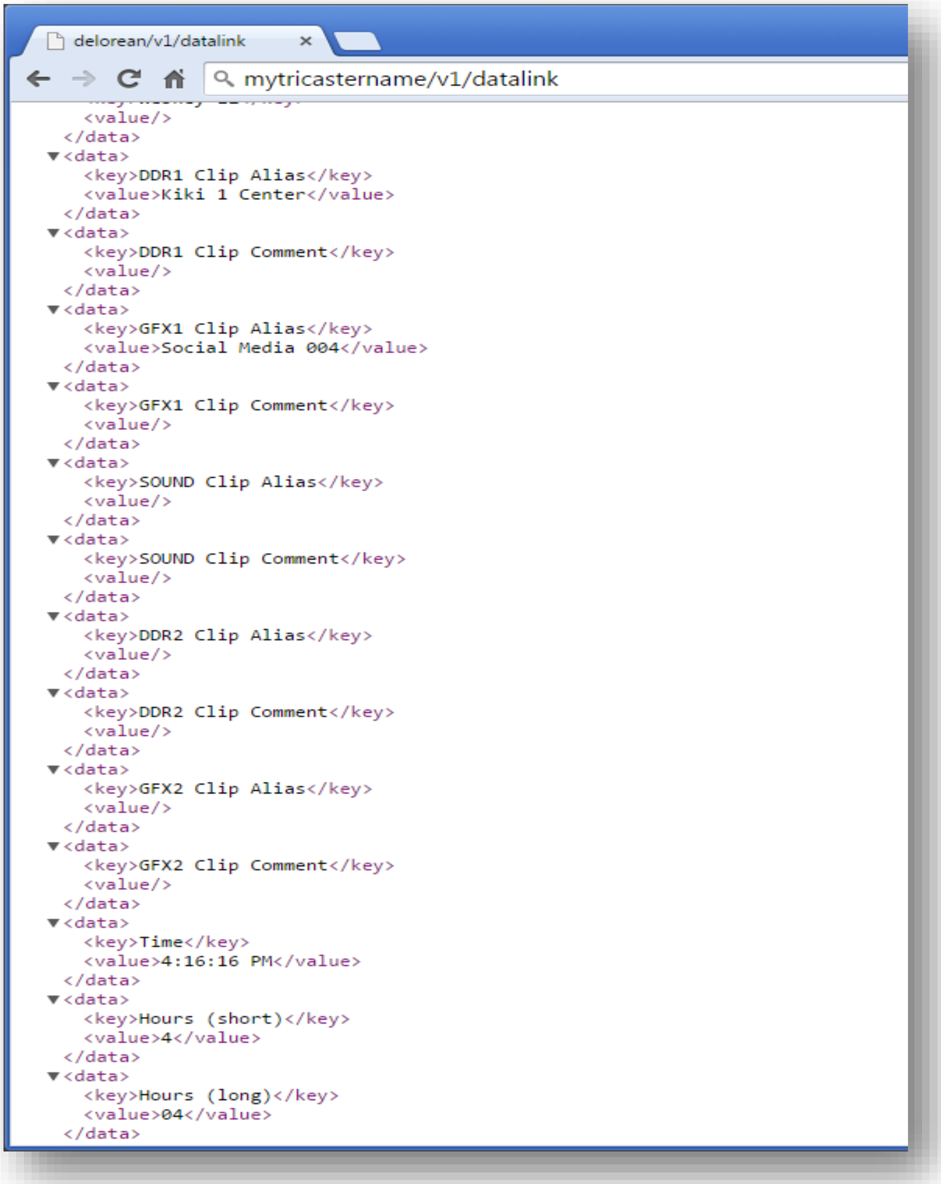
/v1/shortcut?name=NAME&value=VALUE&value1=ANOTHER_VALUE

To issue a trigger you would simply GET the address:

/v1/trigger?name=NAME

To issue a DataLink update, you would GET the address:

/v1/datalink?key=KEY&value=VALUE



```
    <value/>
  </data>
▼<data>
    <key>DDR1 Clip Alias</key>
    <value>Kiki 1 Center</value>
  </data>
▼<data>
    <key>DDR1 Clip Comment</key>
    <value/>
  </data>
▼<data>
    <key>GFX1 Clip Alias</key>
    <value>Social Media 004</value>
  </data>
▼<data>
    <key>GFX1 Clip Comment</key>
    <value/>
  </data>
▼<data>
    <key>SOUND Clip Alias</key>
    <value/>
  </data>
▼<data>
    <key>SOUND Clip Comment</key>
    <value/>
  </data>
▼<data>
    <key>DDR2 Clip Alias</key>
    <value/>
  </data>
▼<data>
    <key>DDR2 Clip Comment</key>
    <value/>
  </data>
▼<data>
    <key>GFX2 Clip Alias</key>
    <value/>
  </data>
▼<data>
    <key>GFX2 Clip Comment</key>
    <value/>
  </data>
▼<data>
    <key>Time</key>
    <value>4:16:16 PM</value>
  </data>
▼<data>
    <key>Hours (short)</key>
    <value>4</value>
  </data>
▼<data>
    <key>Hours (long)</key>
    <value>04</value>
  </data>
```

FIGURE 34

You can easily obtain a full list of the current set of DataLink key/value pairs by querying the URL:

http://*TriCasterSystemName* /v1/datalink

This returns the current DataLink keys and values (Figure 34) in the format below:

```
<datalink_values>
    <data>
          <key>Time</key>
          <value>5:08:50 PM</value>
  </data>
  <data>
          <key>Hours (short)</key>
          <value>5</value>
  </data>
  <data>
          <key>Hours (long)</key>
          <value>05</value>
  </data>
  <data>
          <key>Hours (short, 24hr)</key>
          <value>17</value>
  </data>
  <data>
          <key>Hours (long, 24hr)</key>
          <value>17</value>
  </data>
  <data>
          <key>Minutes (short)</key>
          <value>8</value>
  </data>
Etc…
```

Note: The name and value pairs must be correctly escaped.

8.4.2 POST COMMANDS

You can POST a <shortcut …> or <trigger …> message directly. You should send these to any URL that is in the path /v1/shortcut, /v1/trigger or /v1/datalink. Since these are XML messages, the content-type should be text/xml. For instance, if you wished to send a shortcut contained in the file "MyShortcut.xml", you would use the following CURL command line:

```
curl -X POST -d @MyShortcut.xml http://TriCasterSystemName/v1/shortcut --header "Content-Type:text/xml"
```

If you are using PHP, it is common to wish to post commands in a form "shortcut=<shortcut name=value…". This is automatically detected for "shortcut=", "trigger=", "datalink=" where the value is the regular XML form of the shortcut. The value is full escaped.

8.4.3 FILE TRANSFER

It is possible to send media files directly to a TriCaster via HTTP post commands. Post commands are sent to http://TriCasterSysemName/v1/file. As an example, the following curl command would post a PNG file to the TriCaster system.

```
curl.exe -X POST –data-binary @YourCoolFile.png http://TriCasterSystemName/v1/file --header "Content-Type:
image/png " --header "Filename:HelloWorld.png" --header "Overwrite:true"
```

The header "Filename:somefilename" is optional, but allows you to give the TriCaster a hint as to what filename to choose for this file. If one is not specified then a filename will be chosen for you automatically.

Specify "Overwrite:true" if you wish the filename to be used no matter whether it exists or not (by default, overwrite is false).

The response from the file transfer will either be a 400 error, with a text description of the error, or a 200 success with the filename of the file returned; allowing you to then use this in titles, DDRs, etc.

Supported content types are:

Type	Description
audio/mpeg	MP3 Audio file
audio/basic	AU Audio file
audio/x-wav	WAV file
audio/x-aiff	AIFF file
image/gif	GIF image file
image/jpeg	JPEG image file
image/png	PNG image file
image/tiff	TIFF image file
image/bmp	BMP image file
video/x-msvideo	AVI file
video/quicktime	QuickTime file (MOV)
video/mp4	MPEG-4 wrapper (H.264 normally)
video/mpeg	MPEG-2 video file.

8.4.4 VIDEO PREVIEWS

It is possible to receive JPG images back for any source within the TriCaster or 3Play system. These may be queried as described next.

Possible names of sources for use with this method are listed in the table below:

Source Name	Description	Products
output1	The primary video output	All TriCaster units
output2	The secondary video output	All TriCaster units
output3	The third video output	TC8000 only
output4	The fourth video output	TC8000 only
input1– input8	Video inputs	All TriCasters (input1-4 only on some models)
net1, net2	Network inputs	All TriCaster units
ddr1, ddr2	DDR media players	All TriCaster units
gfx, gfx2	Graphics players	All TriCaster units
bfr1 – bfr15	15 buffers	All TriCaster units

Additional parameters are available that perform the following:

Parameter	Description
xres	Force the resolution of the image to be this width. If this is omitted, this is computed from the source aspect ratio.
yres	Force the resolution of the image to be this width. If this is omitted, this is computed from the source aspect ratio.
q	JPEG image quality (range 50-100).

TABLE 1

In general, it is wise not to query frames at high image rates. Request them at a resolution that is close to what you need. For instance, to receive a 640x480 preview of the current output, you would get:

http:// TriCasterSystemName/v1/image?name=output&xres=640&yres=480

Note: See also Section **Error! Reference source not found.** (**Error! Reference source not found.**) for information on iCaster's WebSocket implementation.

8.4.5 GENERATING ICONS

It is possible to obtain an icon representation for any file that is on the system. This can be used, for instance, to generate icons for LiveSet or Transition files that may be loaded into the switcher (with the names given to you by getting the TriCaster state data). Similar to how inputs are read, you get an icon by specifying the path (required), the resolution of the longest side (optional), and the JPEG quality (optional).

For instance to get the icon for the file:

C:\TriCaster\Effects\LiveSets\NewTek\Alignment\Center.LiveSet

You could go to the URL :

http://TriCasterSystemName/v1/icon?filename=C:%5CTriCaster%5CEffects%5CLiveSets%5CNewTek%5CAlignment%5CCenter.LiveSet&res=1920&Q=90

8.4.6 TALLY AND SETTINGS DATA

TriCaster can provide XML-formatted lists of parameters and values representing various states of the switcher, buffers and effects. Combining this information with scripted shortcut commands, sophisticated interactions can be made between the TriCaster and a web application, controlled from either side of the client/server relationship.

Request the URL below (where "TriCasterSystem" is the system name), returns a full description of all of the video inputs on the system, with information regarding whether they are currently being displayed on program or preview output (tally information):

http://TriCasterSystemName/v1/dictionary?key=tally

An example returned XML result would be as follows:

```
<tally>
  <column name="input1" index="0" on_pgm="false" on_prev="false"/>
  <column name="input2" index="1" on_pgm="false" on_prev="false"/>
  <column name="input3" index="2" on_pgm="false" on_prev="false"/>
  <column name="input4" index="3" on_pgm="false" on_prev="false"/>
  <column name="input5" index="4" on_pgm="false" on_prev="false"/>
  <column name="input6" index="5" on_pgm="false" on_prev="false"/>
  <column name="input7" index="6" on_pgm="false" on_prev="false"/>
  <column name="input8" index="7" on_pgm="false" on_prev="false"/>
  <column name="net1" index="8" on_pgm="false" on_prev="false"/>
  <column name="net2" index="9" on_pgm="false" on_prev="false"/>
  <column name="ddr" index="10" on_pgm="false" on_prev="false"/>
  <column name="ddr2" index="11" on_pgm="false" on_prev="false"/>
  <column name="gfx1" index="12" on_pgm="true" on_prev="false"/>
  <column name="gfx2" index="13" on_pgm="false" on_prev="false"/>
  <column name="bfr1" index="14" on_pgm="false" on_prev="false"/>
  <column name="bfr2" index="15" on_pgm="false" on_prev="false"/>
  <column name="bfr3" index="16" on_pgm="false" on_prev="false"/>
  <column name="bfr4" index="17" on_pgm="false" on_prev="false"/>
  <column name="bfr5" index="18" on_pgm="false" on_prev="false"/>
  <column name="bfr6" index="19" on_pgm="false" on_prev="false"/>
  <column name="bfr7" index="20" on_pgm="false" on_prev="false"/>
  <column name="bfr8" index="21" on_pgm="false" on_prev="false"/>
  <column name="bfr9" index="22" on_pgm="false" on_prev="false"/>
  <column name="v1" index="23" on_pgm="false" on_prev="false"/>
  <column name="v2" index="24" on_pgm="false" on_prev="false"/>
  <column name="v3" index="25" on_pgm="false" on_prev="false"/>
  <column name="v4" index="26" on_pgm="false" on_prev="false"/>
  <column name="v5" index="27" on_pgm="false" on_prev="false"/>
  <column name="v6" index="28" on_pgm="false" on_prev="false"/>
  <column name="v7" index="29" on_pgm="false" on_prev="false"/>
  <column name="v8" index="30" on_pgm="false" on_prev="false"/>
  <column name="bfr10" index="31" on_pgm="false" on_prev="false"/>
  <column name="bfr11" index="32" on_pgm="false" on_prev="false"/>
  <column name="bfr12" index="33" on_pgm="false" on_prev="false"/>
  <column name="bfr13" index="34" on_pgm="false" on_prev="false"/>

  <column name="bfr14" index="35" on_pgm="false" on_prev="false"/>

  <column name="bfr15" index="36" on_pgm="false" on_prev="false"/>

  <column name="black" index="41" on_pgm="false" on_prev="false"/>

</tally>
```

Other system information can also be obtained in this manner. For example, requesting the (example) URL below returns a wealth of switcher parameters.

```
http://TriCasterSystemName/v1/dictionary?key=switcher
```

The result would include:

- All inputs, physical and virtual
- Switcher row sources for M/Es, including up to 4 rows if using a LiveSet
- Overlay information for main and M/E switchers
  - Overlay source
  - T-Bar position
- Currently loaded Effect (Transition or LiveSet)

An example response would look like the following:

```
<switcher_update main_source="BFR4" preview_source="gfx1"
effect="Q:\Products\TriCaster_Content\Animation Stores\Output\Broadcast\Door Slam.effect">

 <tbar position="0.000000" speed="0.000000"/>

 <switcher_overlays>

  <overlay z_order_position="0" source="V2">

   <tbar position="0.000000" speed="0.000000"/>

  </overlay>

  <overlay z_order_position="1" source="BFR1">

  <tbar position="0.000000" speed="0.000000"/>
```

Or, requesting the URL below will return a list of the currently-assigned buffers for the main switcher row and the M/E rows, similar to the following example:

```
http://TriCasterSystemName/v1/dictionary?key=buffer

:

<buffers>

 <main>

  <buffer selection="BFR4"/>

 </main>

 <me index="0">

 <row>

  <buffer selection="BFR1"/>
```

</row>

</me>

…

The URL below returns a list of all the effects loaded into all the bins for each transition and overlay in the main Switcher and all M/Es.

http:// TriCasterSystemName/v1/dictionary?key=switcher_ui_effects

An example result follows:

```
<switcher_ui_effects>
  <switcher name="main">
    <effect_bin>
      <effect0 effect="Fade"/>
      <effect1 effect="Q:\Products\TriCaster_Content\Animation Stores\Output\Broadcast\Door
      Slam.effect"/>
      <effect2 effect="c:\TriCaster\Effects\Transitions\Fades\Non Additive Fade.trans"/>
      <effect3 effect="c:\TriCaster\Effects\Transitions\Fades\Flash.trans"/>
      <effect4 effect="c:\TriCaster\Effects\Transitions\Fades\Clouds.trans"
...
      <effect7 effect="c:\TriCaster\Effects\Overlays\Iris\Hard\Rectangle (H).ofx"/>
      <effect8 effect="c:\TriCaster\Effects\Overlays\Iris\Hard\Circle(H).ofx"/>
    </effect_bin>
  </key>
</switcher>
<switcher name="v3">
  <effect_bin>
    <effect0 effect="Fade"/>
    <effect1 effect="c:\TriCaster\Effects\Transitions\Fades\Additive Fade.trans"/>
    <effect2 effect="c:\TriCaster\Effects\Transitions\Fades\Non Additive Fade.trans"/>
    <effect3 effect="c:\TriCaster\Effects\Transitions\Fades\Flash.trans"/>
    <effect4 effect="c:\TriCaster\Effects\Transitions\Fades\Clouds.trans"/>
    <effect5 effect="c:\TriCaster\Effects\Transitions\Fades\Noise.trans"/>
    <effect6 effect="c:\TriCaster\Effects\Transitions\Iris\Hard\Circle(H).trans"/>
...
<switcher name="v8">
  <effect_bin>
    <effect0 effect="C:\TriCaster\Effects\LiveSets\User\Rock Stage Standing Flares 03
    RAW\Rock Stage Standing Flares 03 RAW.LiveSet"/>
  </effect_bin>
  <key>
    <effect_bin>
      <effect0 effect="Fade"/>
      <effect1 effect="c:\TriCaster\Effects\Overlays\Trajectories\Fly In\Fly In B.ofx"/>
  ...
```

The complete list of key values supported for use in the manner disclosed in the preceding examples follows below:

- shortcut_states
- tally
- ddr_playlist
- ddr_timecode
- buffer
- switcher
- buffer
- switcher_ui_effects

8.4.7 TRICASTER WEBSOCKETS

The WebSocket protocol is supported by most major web browsers and allows Tricaster Advanced Edition and 3Play Mini to push data (inluding a great deal of system status information along with image previews and audio data) to the client. This allows improved responsiveness by notifying clients of state changes, removing the need for clients to peridically poll for changes.

The WebSocket implementation, including information on connecting to TriCaster and retrieving data, is discussed in full in the documents available to members of NewTek's Developer Network.

Chapter 9 FILES AND STORAGE

Some corollary of 'Murphy's Law' must state that "The more critical it is to a production, the greater the likelihood that important media will be delivered at the last possible moment and the wrong format, if it can be found at all."

This section is devoted to reducing your stress level when this inevitably occurs.

Section 9.1 MEDIA FILE FORMATS

9.1.1 VIDEO CAPTURE

Current TriCaster products support a number of different file formats for capture, including several native Quicktime™, AVI, MPEG-2 and H.264 formats. In general, where other considerations permit, we recommend the use of one of the two Quicktime formats for video capture.

These two file types support all TriCaster features, including embedded timecode, and provide high quality capture suitable for almost any purpose (the 4:2:2 format has an edge if you are planning on chromakeying the recorded footage later). Generally, most modern software applications can work with at least one of these Quicktime formats.

Note: TriCaster and 3Play are able to play back Quicktime files that are still 'growing' (being actively captured). Some external software applications can enjoy the same benefits for files being recorded on NewTek systems in Quicktime® format by supporting NewTek's File Reader SDK, available to members of the NewTek Development Network, or by accessing recordings underway using TriCaster's MPEG-2 encoding.

9.1.2 NEWTEK CODECS

Codecs for both Windows® and OS X® systems are available for NewTek's high quality AVI and Quicktime® capture formats. The codecs are included in the "Extras" folder of all NewTek live production systems, and can alternatively be downloaded from the Support pages of the NewTek website.

WINDOWS® PLATFORM

Users of Microsoft Windows® systems (other than TriCaster or 3Play) will need to install NewTek's Quicktime™ codecs and, of course, the Apple Quicktime® player to be able to read and write files in these formats .

The NewTek codec pack for Windows also provides read and write support for NewTek's proprietary SpeedHQ AVI format to suitable video applications.

OS X® Platform

The NewTek Quicktime codecs are not required to be able to read TriCaster or 3Play Quicktime® files on OS X systems. However – depending on what other software you already have installed – you may need to download and install a 'professional' codec pack available from Apple® at the ULR shown below (at the time of writing):

http://support.apple.com/kb/DL1396

Apple® platform users may find it useful to install the NewTek Codec Pack for OS X® anyway, since it does offer certain added benefits. Specifically, suitable applications (such as Compressor™) will be able to access them to read and write NewTek's SpeedHQ Quicktime® files, and even read the SpeedHQ AVI file format.

Section 9.2 Import

TriCaster and 3Play are able to play back media files in many, many different popular file formats, but it must be admitted that some require more system resources to play than others. With a view to making best use of precious resources, then, ideally media files should be prepared beforehand.

With the assistance of the NewTek codec packs just mentioned, it is often true that files can be prepared in NewTek-friendly formats that are easy to play back right in your favorite non-linear editor or compositing package. We can recommend NewTek's own SpeedHQ Quicktime® encoding as a good high quality format for use in any of our live production systems. (SpeedHQ options include support for files with embedded alpha channel, especially valuable for animations intended for use as overlays.)

Alternatively, dedicated *Import* modules are provided in all modern NewTek live production systems. This module, found in the *Startup* control panel (see the manual for your specific product for details) provides a way to add multiple items to a queue for batch processing, including optional transcoding as necessary.

Otherwise, most high quality Quicktime formats (other than ProRes) will work reasonably well. For HD files, you might consider trying the Quicktime PNG encoder (especially when an alpha channel is required).

Section 9.3 Export

At times you may wish to export files recorded with a NewTek live production in some other popular format. Of course, whether working with files captured to shared storage systems or copied to external media across a network, for example, you could perform transcoding entirely externally using your favorite conversion software. You may instead, though, wish to use your NewTek system to do transcoding.

All current NewTek live production platforms include an Export feature in the system's Startup pages. Most also provide a Publish Queue in the Live or Replay Desktop. Files can be added to the lists in these modules using a variety of method, even – in this latter case – during live production. The modules provide access to a deep set of transcoding tools and control over destinations for file output (including local and networked volumes, and ftp).

Section 9.4 Asset Management

The integrated Media Browser native to NewTek live production systems is a competent asset management system, enabling to quickly locate and work with files related to your sessions, or external files. Of course, more extensive media asset management systems provided by leading industry providers are also available within the NewTek ecosystem, and may be directly supported by NewTek products.

To utilize your favorite (supported) third-party asset management systems, you need simply hold down the keyboard Ctrl key when invoking a file browser. For example, double-clicking a blank spot in a DDR playlist on TriCaster with Ctrl depressed will show your compatible custom asset management interface, rather than TriCaster's native Media Browser.

Hint: Alternatively, you can open a standard system file explorer, by holding down the Shift key rather than Ctrl when adding files.

A list of known third-party solutions on offer at the time of writing, along with brief details, will follow (in Developer Network). Asset management solutions can include outboard storage systems, so let's talk about this related matter.

Section 9.5 External Storage

Generally, NewTek live productions systems provide substantial integrated storage for media used in your productions, by means of internal and removable drives. Of course, many broadcast settings have still larger requirements, making external storage solutions attractive. In addition, by virtue of their potential for great capacity, fails-safe mechanisms, transfer speed, and shared access, external storage solutions can facilitate file ingest, shared access, media updates, and more.

Large storage solutions come in many varieties, including SAN (Storage Area Network), NAS (Networked Attached Storage) and others. Individual solutions may include dedicated MAM (Media Access Management) implementations, or not.

In general, we recommend the use of the NTFS file system, not least because (unlike, for example, FAT32) it properly handles files larger than four gigabytes, but also because it fully supports TriCaster and 3Play features.

At times, though, you may prefer to employ another file system for media used for video capture or file sharing. If so, note that it's best if the actual "Session Volume" for TriCaster or 3Play is still NTFS-formatted. Otherwise, links to media captured during the session that are automatically generated by the system may fail, forcing you to expend extra effort to locate them.

APPENDICES

APPENDIX A.DEVELOPER NETWORK

A.1 OVERVIEW

The NewTek Developer Network is an ongoing initiative, and a key cornerstone of our open platform strategy.

The purpose of the NewTek Developer Network is to provide developers with the tools they need to create applications and products that integrate with and augment the NewTek video products, particularly the TriCaster™ or 3Play™ lineup. The program provides access to and support for various SDK/API components prepared for diverse implementations.

A.2 SOFTWARE DEVELOPMENT KIT

Some existing level of programming expertise is required to use most of the components included in the NewTek Software Development Kit. Each component is fully documented in this distribution, and may be accompanied by Example and guidelines useful in creating applications or plug-ins to support and integrate with NewTek products.

A.3 PARTICIPATION

A partial listing of current Developer Network membership can be accessed at the URL below:

http://www.newtek.com/solutions/newtek-developer-network.html

To apply to participate in the NewTek Developer Network program, please send an email with your full contact details to: videosdk@newtek.com.

APPENDIX B.THIRD PARTY SOLUTIONS

B.1 OVERVIEW

Our development partners have prepared a wide array of powerful hardware *and* software solutions for use with NewTek live production systems.

The information that follows is not exhaustive, and your system integrator of NewTek support channel is always able to provide much more up to date and comprehensive information. Nevertheless, we wanted to give you a little taste of what's available.

Likely that you will find more than one product in the pages that follow that fills your particular workflow and pipeline requirements, and perhaps others that inspire you with creative possibilities. Certainly those offering solutions related to your needs will also doubtless be glad to both field questions and hear your suggestions.

B.2 GRAPHICS AND CONTENT

Just as 3Play and TriCaster can interact, sending both audio and video signals as well as control commands between network connected systems and software, many third-party developers leverage various SDKs to provide extensive communication and data exchange capabilities throughout the NewTek ecosystem. Likewise, network tally can be supported in this manner.

Many products directly supporting AirSend™ respond to the commands detailed earlier in this chapter (Section 8.1). Third-party developers are also able to define their custom commands for execution within the NewTek Macro system (the documentation supplied with your third-party product details custom commands of this type). Alternatively, some third-party products use other available methods of communication with NewTek systems.

A brief introduction to solutions from some third-party providers belonging to the NewTek Development Network available at the time of writing follows below. Generally the solutions mentioned in this section provide graphics, animation and video transmission, supplemental external control, or a combination of both.

B.2.1 AGF MULTIMEDIA

CharacterWorks is a character generator and motion graphics application based-on a state-of-the-art, real-time, multi-core, GPU-based, Full-HD, true 3D rendering engine, with smooth output over the network to NewTek live production systems.

CharacterWorks supports playback of HD image sequences, dynamic text with embedded animations and data from shared external sources in real-time, all in an affordable, integrated package. Supported sources include Twitter, XML (including RSS) through XQuery, and external databases through ODBC/SQL (including MySQL, Access, Excel).

The Automation Editor allows you to schedule behaviors, including playback of CW graphics elements or remote invocation of Tricaster functionality, at specified wall-clock times. CharacterWorks graphics can be edited and triggered remotely from TriCaster, allowing a single operator to control both TriCaster and CW.

For further information, visit http://chrworks.com

B.2.2 BRAINSTORM MULTIMEDIA

Brainstorm's AirSend implementation allows integration of a wide range of Brainstorm's real-time 3D graphics and CG products, including real-time 3D graphics and virtual set solutions, with NewTek live production systems.

High quality, sophisticated content produced by Brainstorm Aston 3D graphics creation, CG and playout application can be streamed live over an IP connection to NewTek live production systems for inclusion in and to enhance program material.

Brainstorm's Infinity Set product, with NewTek AirSend support planned, offers a revolutionary approach to virtual set production, including TrackFree technology which allows combining tracked and trackless techniques in the same camera.

For further information, visit http://brainstorm.es/products

B.2.3 FINGERWORKS TELESTRATORS

Telestration tools are extremely valuable for news, sports, and weather productions, and live presentations of all sorts. FingerWorks™ telestration products provide foreground/background separation (place graphics and animations *under* players), scalable animations, 3D tools, particle effects and more. FingerWorks 5 takes advantage of the network video i/o capabilities of TriCaster or 3Play, simplifying connections and work flow.

For further information, visit http://telestrator.com

B.2.4 VIZRT

Integrated with NewTek live production systems, Vizrt's character generator Viz Trio provides easier control of live video and graphics content in the control room, OB truck or stadium for use on-air or online.

Vizrt also offers packages integrating its IP-based graphics production tools with NewTek live video production systems. Helping to create a smarter workflow for broadcasters, the package provides easier control of live video and graphics content in the control room, OB truck or stadium for use on-air or online, utilizing its IP-based Viz Engine and Viz Trio as part of an integrated studio production system. The IP-stream is recognized automatically as a source for graphics overlay, in this single-box solution with animated preview and bundled Viz Engine.

For further information, visit http://www.vizrt.com/products/viz_trio

B.2.5 CASPARCG

CasparCG was developed by the Swedish Broadcasting Corporation ("SVT"), which is the license-fee funded public service television broadcaster in Sweden. What started as an in-house tool quickly developed into powerful open source software allowing graphics and video to be sent over the network NewTek live production system for direct use in production.

CasparCG supports all standards and plays all common formats. It can play simultaneous layers of dynamic graphics, videos and images in real-time for all broadcast, show, event and signage needs. Wide codec support includes ProRes, DNxHD and DVCPROHD for both playback and rendering to disk.

For further information, visit http://www.casparcg.com

B.2.6 LIVEXPERT

LiveCG Broadcast from LiveXpert features static and animated graphics and logos, dynamic text, clock, date, crawl, ticker and roll. It plays TGA, BMP, PNG, TIF, JPG, GIF sequences and FLASH animations, and supports shadow, blur, motion blur, and edge smoothing, along with transitions: fade, move, and zoom. The page composer provides multiple layers, and the system supports GPI on TCP/IP (Optional RS-232).

LiveCG Football is a professional graphics management application for NewTek production systems. It provides complete management of live football productions, including scoring, timing, statistics, and database. LiveCG Football runs from a remote computer connected to the network input of a NewTek live production system.

For further information, visit http://www.3dstorm.com

B.2.7 MEDIA 5

CG5 NET IP® is a multilayer character generator providing a complete, high quality and versatile broadcast graphics solution suitable for any production company.

This tool can supply real time titles and subtitles, including branding, logos, lower thirds, crawls, rolls, scaled videos and much more. It supports unlimited layers with separate speed and fade in/out controls, multiple 3D animated logos display with separated controls, time and weather display, sports timers, multiple scrolls and crawls with separate controls, and multiple video insertions (with alpha channels).

For further information, visit http://video5.tv/en/portfolio/cg5-net-hd/

B.2.8 GRAPHICS OUTFITTERS

Graphics Outfitters' IVGA Connect product allows connection of any external HD-SDI source directly over the network to NewTek live production systems. It utilizes the full available bandwidth to losslessly pass video from the HD-SDI source (including key and fill) without tying up any video inputs.

ScoreHD is the perfect solution for scorebug creation and playout with real-time rendering of transitions and motion effects. Sport specific software provides operator ease of use with custom layouts, animation and glow/blur effects.

SketchIVGA is a complete 1RU telestration system. Just feed your "clean-feed" HD-SDI program out feed into the SketchIVGA input for background display on any supported Touchscreen monitor. Telestration key and fill outputs presented to the NewTek system's network inputs.

For further information, visit http://graphicsoutfitters.com

B.2.9 COMPIX

Compix Linx™ is a software-based, fully-featured character generator with quality perfected over 26 years from 20,000 channels around the world. With no proprietary CG hardware to buy, users can produce sophisticated on-screen graphics using a laptop or desktop PC. Compix Linx also offers channel branding capabilities with automatically updating headline feeds, and the ability to broadcast social media conversations from Twitter and Facebook. This graphics powerhouse includes over 260 transition effects, unlimited true type fonts, image and animation import, spell checker, and multilingual support.

For further information, visit http://www.compixlinx.com

B.2.10 CHYRONHEGO

ChyronIP is a real-time HD/SD 2D and 3D character and graphics generator specifically designed for NewTek live production systems. Leveraging the company`s award-winning Lyric PRO graphics application, ChyronIP provides producers with up to two full motion HD or SD channels of Chyron graphics (single-channel standard, second channel optional) that stream directly over a network connection without tying up video inputs.

For further information, visit http://chyronhego.com/broadcast-graphics/chyronip

B.2.11 CLASSX

With ClassX solutions for CG and graphics, TriCaster users are able to create their media playlist and synchronize graphics and data with the ease of a modern and innovative user interface with ContentPlayout, a new product add-on for the ClassX LiveBoard family. For sports, news, games, entertainment, social media and more, ClassX products are affordable and completely integrated with NewTek products through the IP network.

For further information, visit http://www.classx.it/en/broadcast-products.html

B.2.12 TOFERVISION

LT [Scoreboards] provides 14 new skins for NewTek live production systems with LiveText2 and greatly simplifies the operation of scoreboard graphics for a large variety of sports productions.

For further information, visit http://tofervision.com/software/ltscoreboards/

B.2.13 APRIL BROADCAST

Browse, select, buy and immediately get a link to download a premium virtual set (whether as a layered .psd file for use with NewTek's Virtual Set Editor™ application or as a LiveSet™ ready for immediate use (see samples shipping with TriCaster). Custom-build services are also available.

For further information, visit http://aprilbroadcast.tv/tricaster

B.2.14 VIRTUALSETWORKS

Virtual sets for TriCaster expand your virtual set offerings and allow for editing and customization via NewTek's Virtual Set Editor™ or direct utilization within the switching environment.

For further information, visit http://www.virtualsetworks.com

B.2.15 VIRTUALSETS.COM

Custom virtual set designs provide users with a professional, highly realistic scenic environment. VIRTUALSETS.COM designs fully custom virtual sets and re-brands existing designs for their clients.

For further information, visit http://virtualsets.com

B.2.16 FREEPLAY MUSIC

Freeplay plans to display its music search engine and automated license process right on the desktop for NewTek TriCaster users, making production with great music simple and easy.

For further information, visit http://tricaster.freeplaymusic.com/

B.3 MAM & Storage Solutions

What follows are brief details of some third-party solutions available at the time or writing from providers belonging to the NewTek Development Network.

Note: Generally, the solutions discussed in the next section require a TriCaster 410, 460, 860 or 8000, or a 3Play 4800 or 440. See the provider's websites for more detail.

B.3.1 VITEC

Proxsys PA TC Video Archiver is a powerful solution to manage and archive NewTek TriCaster sessions and recordings. By supporting both network and removable disk file exchange, the solution provides online and LTO 5/6 tape management for universal workflows, integrated file preview generation, and cost effective media archive.

The system employs an easy to use browser-based UI. Keeping the previews online allows the user to view all content, while the Hi-Res files are stored on cost effective and reliable LTO media. The customized metadata model, automated metadata import from TriCaster and the sophisticated search engine allow you to manage large archives.

For further information, visit:

http://www.vitec.com/products/video-management/archiving-isr-fmv-nle/product/show/PA-105060

B.3.2 STUDIO NETWORK SOLUTIONS (SNS)

When using the EVO Shared Storage System, users of NewTek live production systems can capture up to 8 camera ISOs directly to the EVO storage server, while playing back content such as stills, clips, and audio from EVO without limiting other network sources.

For further information, visit http://www.studionetworksolutions.com

B.3.3 EDITSHARE®

EditShare® shared storage solutions provide NewTek live production systems with a well-integrated live production and shared storage platform offering extensive media asset management and archiving capabilities.

EditShare's powerful Flow media asset management layer controls NewTek content from capture to post-event production, and back to playout. Coupled with EditShare Ark, Flow provides total control over 'nearline' and offline backup and archiving to spinning disks and/or LTO tapes.

For further information, visit http://editshare.com/products/storage-options

B.3.4 PROMAX SYSTEMS

NewTek live production systems can record up to 8 streams of HD (16 streams from multiple systems) to the Platform Shared Storage System, extending total recording time and flexibility. This integration allows you to store between 32TB - 256TB of content and simultaneously play back content stored on the server giving you access to an unlimited library of content.

For further information, visit http://www.promax.com/s-273-platform-technology-partners.aspx?#Newtek

B.3.5 AXLE VIDEO

axle 2014, the latest edition of axle's award-winning media management software., works with almost any type of storage and folder structure. axle 2014 automatically takes care of discovering your media. Just point it at your existing file system and it indexes your files. As new files are added or removed, axle updates its database in real time. It also automatically creates low-bandwidth proxies you can access from any web browser, imports metadata from your media, and lets you add your own custom metadata as well.

Using a simple browser interface, it's easy to search, comment, mark, approve, and annotate your assets from any location. There's no need to move your media files or change your system setup. Your storage can be a SAN, a NAS or just a local RAID, and can be running in a Mac, Windows or Linux environment. (For optimal performance, it is recommended that axle runs on a dedicated Mac Mini.) Export media, with marking and comments, directly into Adobe Premiere Pro, Apple Final Cut Pro, or Avid Media Composer (Pro configuration).

For further information, visit http://axlevideo.com

B.3.6 SQUARE BOX SYSTEMS

CatDV, from Square Box Systems, is a powerful asset manager that runs directly on NewTek live production systems. By holding down the Ctrl key when opening the Media Browser, operators can search the CatDV database and bring assets into the the system for play out etc. (The CatDV plug-in for NewTek systems is available as a free of charge update for those with CatDV Web licensees).

For further information, visit http://www.squarebox.com

B.4 AUTOMATION

B.4.1 NEWSMAKER SYSTEMS

NewsMaker Systems uses the MOS protocol to provide seamless integration between the TriCaster and Broadcast Newsroom Computer System rundowns. Whole shows are uploaded and changes dynamically made in real-time, relieving the TriCaster operator from the task of managing the show's media material.

For further information, visit http://newsmakersystems.com/products/LANLINK AB

B.4.2 RUNDOWN CREATOR

Rundown Creator is a web application for TV, radio, and internet broadcasters who need an easy way to collaboratively create rundowns for their shows, script them out, and time them. With its TriCaster integration, now you can insert titles, graphics, audio clips, and video clips into your scripts in Rundown Creator, and play them out on your TriCaster using Rundown Creator's new TriCaster Playout Controller.

For further information, visit http://rundowncreator.com

B.4.3 AP ENPS

AP ENPS integrates with TriCaster via the NewsMaker MOS gateway, for publishing content to broadcast and digital media platforms.

For further information, visit http://enps.com

B.4.4 YOUNGMONKEY

This developer offers a wide range of products to enhance just about every NewTek system, and to better integrate them with third party products. The Master Control system can streamline workflow, providing superb functionality and ease of use for operators.

For further information, visit http://www.youngmonkey.ca

B.5 Hardware Controllers

B.5.1 LANLINK

Lanlink SHOUT connects easily to NewTek systems using a USB port, and provides sequential macro playback to help you automate your show. Their MIDIout plugin lets operators connect and control outboard equipment with MIDI support using macros executed from NewTek systems. (Lanlink also offers a tally solution.)

For further information, visit http://lanlink.tv

B.5.2 X-KEYS

X-keys clearly labeled, dedicated keys can streamline your workflow. Any key can be programmed to trigger any command and keys can be organized in logical groupings to fit your workflow.

For further information, visit http://piengineering.com/xkeys.php

B.5.3 LIVEXPERT

LiveMixer from LiveXpert is a versatile application that provides remote control over the audio mixer of TriCaster Professional series systems. LiveMixer supports the affordable Berhinger BCF2000 and the professional Yamaha 01V96i audio consoles.

For further information, visit http://www.3dstorm.com

B.5.4 AVID

Avid Artist Mix provides tactile control over TriCaster's native internal audio mixer, providing hands-on control and precision. Solo, mute, pan, and balance audio sources directly from this hardware control surface, complete with motorized faders.

For further information, visit http://www.avid.com/US/products/artist-mix

B.5.5 TELEMETRICS

Telemetrics Inc. is a leading developer of high-end camera robotics systems, and provides solutions with tight integration with NewTek live production systems.

For further information, visit http://www.telemetricsinc.com

B.5.6 IDCROMVIDEO

MidiDoll provides control over TriCaster's audio from a third-party control surface or a network connected PC.

For further information, visit http://ultimatetv.es/Idioma/html/MidiDoll.html

B.5.7 TALLY LIGHTS

TALLY-LIGHTS

TALLY-LIGHTS provides wired or wireless connectivity of up to eight cameras through TriCaster USB or tally light ports without taking up a camera input. For further information, visit http://tally-lights.com

B.5.8 METASETZ

metaSETZ

metaSETZ is a world-wide leader in tally light systems for the NewTek TriCaster and Livestream Studio. For further information, visit http://www.metasetz.com/

B.6 Ingest and Connectivity

B.6.1 AJA

AJA's KUMO compact SDI routers are available in three configurations offering 32 SDI inputs and outputs, 16 inputs and outputs, or 16 inputs and 4 outputs. The super-compact 1RU and 2RU formats are a perfect fit for any broadcast, production, or post production environment, from mobile sports trucks and edit suites, through to corporate video installations or live theatrical A/V rigs.

For further information, visit http://www.aja.com

B.6.2 UTAH SCIENTIFIC

The UTAH-100/UDS combines the flexibility of a multi-rate digital routing switcher with the economy of a simple distribution amplifier. This modular system is based on I/O modules with 16 ports, and interconnected by a crosspoint fabric that allows any input signal to feed any number of output ports.

For further information, visit http://utahscientific.com/products/utah100uds.php

B.6.3 ENSEMBLE DESIGNS

The BrightEye NXT Router from Ensemble Designs directly supports control by NewTek live production systems with native video router support. It also lets you flexibly configure BNC connectors as inputs or outputs, allowing you to maximize the configuration for your production. BrightEye NXT 's built-in clean switches let you use cameras and other sources that don't have a reference. Various models support diverse connections simultaneously, including HDMI, SDI and fiber optic support, along with many other benefits.

For further information, visit http://www.ensembledesigns.com

B.6.4 DAKTRONICS

TriCaster live production systems, along with NewTek's LiveText CG application and Daktronics scoreboards enable schools to offer an arena sports experience with limited budgets.

For further information, visit http://www.daktronics.com

B.6.5 GNURAL NET, INC.

Integrate up to four Skype™-connected guests in your productions with Gnuron, an HD-SDI based appliance running a sophisticated application built on the native-Skype API, with a feature set and interface optimized for live production environments.

For further information, visit http://www.gnuralnet.com

B.6.6 TERADEK

Wirelessly import live HD video feeds directly into TriCaster from locally connected Teradek Cube encoders or remote bonded cellular devices.

For further information, visit http://teradek.com/pages/cube

B.6.7 MUSHROOM NETWORKS

Mushroom Networks' Webcaster can utilize the 3G/4G bonding appliance Streamer to supply Internet connectivity for high quality and reliable TriCaster streaming.

For further information, visit:

https://www.mushroomnetworks.com/forms/StreamerandTricaster/0,1,1000,1200

B.6.8 WOWZA MEDIA SYSTEMS

Wowza Streaming Engine is robust, customizable, and scalable server software that powers reliable streaming of high-quality video to any device, anywhere. It easily integrates with NewTek's TriCaster to maximize audience's viewing experiences.

For further information, visit http://www.wowza.com/products/streaming-engine

B.6.9 LIVEU

LiveU Link is a software-based solution which allows seamless integration between LiveU's portable uplink units and the TriCaster multi-camera live production system, providing an effortless way to stream live content.

For further information, visit http://liveu.tv/LU70.html

B.6.10 STREAMING MEDIA HOSTING

The SMH Media Platform lets you easily connect, configure and control your TriCaster from any remote location. Streamlined workflow allows for seamless recording, editing, transcoding, and streaming live all from the Cloud to a global audience.

For further information, visit http://streamingmediahosting.com/tools

B.6.11 TELESTREAM

Telestream Wirecast 5 allows you to live stream to multiple platforms or your own streaming servers directly from TriCaster. With Wirecast's pre-configured destinations, it's easy to stream to YouTube Live, Ustream, and many others in just a few clicks.

For further information, visit http://www.telestream.net/#stream

B.6.12 USTREAM

Easier-than-ever HD streaming to Ustream from the TriCaster, no server or encoder configuration required: Login to Ustream and stream with the click of a button directly from TriCaster.

For further information, visit http://www.ustream.tv

B.6.13 HAIVISION

Haivision offers HyperStream Live, a pay-per-use live cloud transcoding software as a service (SaaS), enabling simplified, adaptive delivery of your source video content to set-top boxes, desktops and mobile devices over the Internet. Using the NewTek TriCaster's native encoding, simply connect to your HyperStream Live account and optimized publishing presets are automatically downloaded to your system. When the publishing preset has been loaded and the HyperStream Live transcoder has been started, the

TriCaster operator initiates streaming by simply clicking the Stream button in the TriCaster’s live production Dashboard.

For further information, visit http://www.haivision.com

APPENDIX C. TRICASTER COMMANDS

Most simple shortcuts require prefixes, which are prepended to the shortcut string and identify its target. Shortcuts may or may not take one or more values. Basic *syntax for a simple shortcut is as follows:

```
Prefix_shortcut (value).
```

For example, the "_auto" shortcut requires no value. A prefix (such as "main") tells the shortcut where to apply the related command. Thus the simple shortcut below performs an auto (transition) on TriCaster's main Switcher.

```
main_auto
```

*In use, a shortcut would be formatted as <shortcut name="main_auto"/>

The shortcut below illustrates how a value is supplied. "v1" is the prefix that targets M/E 1. The value "0" identifies the first video input. Thus this shortcut will typically select Camera 1 on the A row of M/E 1.

```
<shortcut name="v1_a_row" value="0"/>
```

Supported prefixes for related groups of shortcuts are listed in tables in the individual sub-headings that follow.

C.1 SWITCHER AND M/E

The shortcuts described in this section control settings and operations affecting TriCaster's main *Switcher* and *M/Es*. Shortcut details are grouped in sub-sections as follows:

A. Background Layers – Configuration and operations involving Program and Preview, or A, B, C D rows
B. Transitions – Effect selection, timing, T-Bar control, etc.
C. Overlays – Configuration and operations involving Key or DSK layers

C.1.1 BACKGROUND LAYERS

Shortcut Prefixes	Description
main, main	With a or b suffix, selects main switcher program or preview row
virtualinputs	All delegated M/Es
v1-v8	M/E specified by number

TABLE 2

Generally, the prefixes described in Table 2 above are valid targets for shortcuts listed in this section. Certain shortcuts may not support every prefix, however. For example, a shortcut whose purpose is to select the source for "Row c" in an M/E will fail with the "main" prefix (the "a" and "b" row alternatives *would* work).

_a_row(int)[state: (int)]

Description: Selects Program or A row source (Value). Valid values start at 0.

Example:

Delay	Shortcut	Value
	main_a_row	0

_b_row(int) [state: (int)]

Description: Selects *Preview* or *B* row source (**Value**). Valid values start at 0.

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	main_b_row	6				

_c_row(int)[state: (int)]

Description: Selects the C row source (**Value**). Valid values start at 0.

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	main_c_row	2				

_d_row(int)[state: (int)]

Description: Selects the D row source (Value). Valid values start at 0.

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	main_d_row	2				

_b_row_named_input(string)[state: (string)]

Description: Selects the Preview or B row source by its internal name (e.g., input1, input2,... ddr, ddr2, stills, etc.)

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	main_b_row_named_input	ddr2				

_a_row_named_input(string)[state: (string)]

Description: Selects the Program or A row source by its internal name (e.g., input1, input2,… ddr, ddr2, stills, etc.)

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	main_a_row_named_input	ddr2				

_c_row_named_input(string)[state: (string)]

Description: Selects the C row source by its internal name (e.g., input1, input2,… ddr, ddr2, stills, etc.)

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	v1_c_row_named_input	input2				

_d_row_named_input(string)[state: (string)]

Description: Selects the C row source by its internal name (e.g., input1, input2,… ddr, ddr2, stills, etc.)

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	V1_d_row_named_input	input1				

_adjust_zoom(double)[state: (double)]

Description: Adjusts the current Animated Zoom duration by the **Value** supplied (in seconds)**.**

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	V1_adjust_zoom	5				

_toggle_animate_zoom(void)

Description: Toggles the LiveSet animation effect between an animated zoom or a cut.

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	v1_toggle_animate_zoom					

_select_zoom_preset(int)

Description: Selects a zoom preset using the supplied **Value** (from 0-7).

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	v1_ select_zoom_preset	5				

_zoom_speed_value(double)

Description: Sets the LiveSet zoom speed *and* performs a zoom. Valid values range from -1.0 to 1.0.

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	v1_zoom_speed_value	1.0				

_panx_speed_value(double)

Description: Sets the LiveSet x-axis pan speed and performs a pan. Valid values range from -1.0 to 1.0.

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	v1_panx_speed_value	1.0				

_pany_speed_value(double)

Description: Sets the LiveSet x-axis pan speed and performs a pan. Valid values range from -1.0 to 1.0.

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	v1_pany_speed_value	1.0				

_set_zoom_duration(double)

Description: Sets zoom duration to the **value** supplied (between 0.0 and 120.0).

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	v1_set_zoom_duration	120				

_set_mix_effect_comp_preset_index(int)

Description: Set preset by index for new composite preset combo in mix effect control. Index starts at 0.

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	v1_set_mix_effect_comp_preset_index	4				

_set_mix_effect_comp_preset_file(index int, filename string)

Description: Choose a file for a slot in new composite preset combo in mix effect control. Index for presets starts at 0.

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	v1_set_mix_effect_comp_preset_file		index	3	filename	C:\TriCaster\Effects\LiveSets\Corner Talk\Basic\Right.LiveSet

_select_preset(int)[state: (int)]

Description: Selects a Switcher or M/E preset (a.k.a. 'MEM slot') specified by the **value** (0-5).

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	v1_select_preset	1				

_copy_preset(int)

Description: Copies a Switcher or M/E preset (a.k.a. 'MEM slot') specified by the **value** (0-5).

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	v1_copy_preset	1				

_paste_preset(int)

Description: Pastes a copied preset into the slot specified by **value** (0-5); the target must be the original M/E.

Example:

Delay	Shortcut	Value
	V2_paste_preset	3

_delete_preset(int)

Description: Deletes the preset specified by **value** in the target/prefix. *For a full listing of valid prefixes see Table 5.*

Example:

Delay	Shortcut	Value
	v1_delete_preset	2

_set_liveset(string)

Description: Load a preset into the target from a path supplied as **value**.

Example:

Delay		Shortcut	Value
		v1_set_liveset	C:\TriCaster\

_select_dockpreset(int)

Description: Select preset at index **value**.

Example:

Delay		Shortcut	Value
		v1_select_dockpreset	2

C.1.2 COMPBINS

_load_compbin(int)

Description: Select compbin by index (**value)**.

Example:

Delay		Shortcut	Value
		v1_load_compbin	3

_save_to_compbin(int)

Description: Save/update to a compbin slot at a specified index (**value)**. Numbering starts at 1.

Example:

Delay		Shortcut	Value
		v1_save_to_compbin	3

_delete_compbin(int)

Description: Delete a slot in the compbin by its index (**value)**. Numbering starts at 1.

Example:

Delay		Shortcut	Value

		v1_delete_compbin	3

_rename_compbin(index int, alias string)

Description: Rename compbin at index (**value)**.

- index – Numbering starts at 1.
- alias – The name you want to apply to the slot (index) in the compbin.

Example:

Delay		Shortcut	Value	Key	Value	Key	Value
		v1_rename_compbin		index	1	alias	NamethisTile

_open_compbin(void)

Description: Open compbin.

Example:

Delay		Shortcut	Value
		v1_open_compbin	

_close_compbin(void)

Description: Select compbin.

Example:

Delay		Shortcut	Value
		v1_close_compbin	

C.1.3 TRANSITIONS

The shortcuts described in this section control settings and operations affecting transitions in TriCaster's main *Switcher* and *M/Es*. Prefixes for use with these shortcuts are listed in Table 3.

Transition Shortcuts	**Description**
main	All main switcher delegates
virtualinputs	Delegated layers for delegated M/Es
v1, v2, v3…	M/E specified by number
main_background	Main switcher background layer
virtualinputs_background	Background layer for delegated M/Es

v1_background, v2_background…	Background layer for ME specified by number
main_dsk1, main_dsk2, main_dsk3, main_dsk4	Main switcher DSK specified by number
virtualinputs_dsk1, virtualinputs_dsk2, virtualinputs_dsk3, virtualinputs_dsk4	Key layer specified by number for delegated M/Es
v1_dsk1, v2_dsk1, v3_dsk1…	Key 1 for M/E specified by number
v1_dsk2, v2_dsk2, v3_dsk2…	Key 2 for M/E specified by number
v1_dsk3, v2_dsk3, v3_dsk3…	Key 3 for M/E specified by number
v1_dsk4, v2_dsk4, v3_dsk4…	Key 4 for ME specified by number
main_ftb	Main switcher, Fade to Black
virtualinputs_ftb	Fade to Black for delegated M/Es
v1_ftb, v2_ftb, v3_ftb…	Fade to Black for M/E specified by number

TABLE 3

_auto(void)[state: (bool)]

Description: Autos between the Program/Preview or A/B rows (optional "bool" controls the "Auto" button light.)

Example:

Delay	Shortcut	Value
	main_auto	

_preview_auto(bool)[state: (bool)]

Description: Performs a preview Auto.

Example:

Delay	Shortcut	Value
	main_preview_auto	

_reversed_auto(void)

Description: Performs a reverse Auto, moving T-bar to the top (compare _goto_top, which jumps to the top).

Example:

Delay	Shortcut	Value

	main_reversed_auto	

_take(void)[state: (bool)]

Description: Triggers a Take (optional “bool” value controls the “Take” button light.)

Example:

Delay	Shortcut	Value
	main_take	

_up(void)

Description: Increments the T-bar position by 1%.

Example:

Delay	Shortcut	Value
	main_up	

_down(void)

Description: Decrements the T-bar position by 1%.

Example:

Delay	Shortcut	Value
	main_down	

_up_fast(void)

Description: Increments the T-bar position by 5%.

Example:

Delay	Shortcut	Value
	main_up_fast	

_down_fast(void)

Description: Decrements the T-bar position by 5%.

Example:

Delay	Shortcut	Value
	main_down_fast	

_goto_halfway(void)

Description: Moves the T-bar to its halfway point.

Example:

Delay	Shortcut	Value
	main_dsk2_goto_halfway	

_goto_top(void)

Description: Jumps the T-bar to the top (compare to _reversed_auto).

Example:

Delay	Shortcut	Value
	v1_dsk2_goto_top	

_goto_bottom(void)

Description: Move the transition T-bar to the bottom.

Example:

Delay	Shortcut	Value
	v1_dsk1_goto_bottom	

_value(double)[state: (double)]

Description: Set the target T-bar to the position specified by the **Value**. Range values from 0 to 1.

Example:

Delay	Shortcut	Value
	main_value	0.5

_toggle_reverse(bool)

Description: Toggles reverse mode of transition (optional "bool" sets a specific state).

Example:

Delay	Shortcut	Value
	main_toggle_reverse	

_toggle_autoreverse(bool)

Description: Toggles "ping/pong" mode of transition (optional "bool" sets a specific state).

Example:

Delay	Shortcut	Value
	main_toggle_autoreverse	

_select_index(int)[state: (int)]

Description: Selects a transition from the Transition Bin slot specified by **Value** (beginning at 0).

Example:

Delay	Shortcut	Value
	main_toggle_autoreverse	

_select_next(void)

Description: Selects the next transition in the target Transition Bin (does not loop after the last item).

Example:

Delay	Shortcut	Value
	main_dsk1_select_next	

_select_prev(void)

Description: Select the previous in transition in the target Transition Bin (does not loop past the first item).

Example:

Delay	Shortcut	Value
	main_dsk2_select_prev	

_select_fade(void)[state: (bool)]

Description: Selects Fade as the active transition.

Example:

Delay	Shortcut	Value
	main_dsk2_select_fade	

_slow(void)

Description: Sets S(low) duration for the current transition or zoom.

Example:

Delay	Shortcut	Value

	main_dsk2_slow	

_medium(void)

Description: Sets M(edium) duration for the current transition or zoom.

Example:

Delay	Shortcut	Value
	main_dsk2_medium	

_fast(void)

Description: Sets F(ast) duration for the current transition or zoom.

Example:

Delay	Shortcut	Value
	main_dsk2_fast	

_speed_next_preset(void)

Description: Cycles through the transition duration presets (Slow, Medium, and Fast).

Example:

Delay	Shortcut	Value
	main_speed_next_preset	

_adjust_speed(double)[state: (double)]

Description: Raises or lowers the transition duration setting by the **Value** entered (in seconds).

Example:

Delay	Shortcut	Value
	main_adjust_speed	-0.5

_speed(double)[state: double]

Description: Sets the duration of a transition to the specified value (in seconds).

Example:

Delay	Shortcut	Value
	main_dsk2_speed	1.5

_tbar_speed(double)

Description: Start an Auto using the transition speed specified by **Value** (in seconds), or update the speed of an Auto already in progress.

Examples:

Delay	Shortcut	Value
	main_dsk2_tbar_speed	1.5

_switch_transition(int)

Description: Selects a new transition in the bin based on a positive or negative offset **Value** from the current slot.

Example:

Delay	Shortcut	Value
	main_dsk2_switch_transition	-2

_set_transition(string)

Description: Load a transition into the active slot from a path supplied as **Value** (delegate-based prefixes, such as “main”, are not supported).

Example:

Delay	Shortcut	Value	Key 0	Value 0
	main_dsk1_set_transition	D:\Path\To\Transition		

_select_saved_nonfade_transition(void)

Description: Selects the last ‘non-Fade’ transition previously used for the designated target.

Example:

Delay	Shortcut	Value	Key 0	Value 0
	main_select_saved_nonfade_transition			

C.1.4 OVERLAYS

The shortcuts described in this section control settings and operations affecting overlays (DSK and Key layers) in TriCaster’s main *Switcher* and *M/Es*.

Prefixes for use with these shortcuts are the same as those for the prior section (Table 3).

_select(int)[state: (int)]

Description: Sets the DSK/Key source to the menu selection defined by index **Value** (starts at 0).

Example:

Delay	Shortcut	Value	Key 0	Value 0
	main_dsk2_select	20		

_select_named_input(string)[state: (string)]

Description: Set the DSK source using internal switcher names as **Value** (e.g., input1, input2, etc.)

Example:

Delay	Shortcut	Value	Key 0	Value 0
	main_dsk1_select_named_input	Ddr2		

_switch_source(int)

Description: Selects a new source based on a positive or negative offset **Value** from the current source (wraps around when the first or last item is reached).

Example:

Delay	Shortcut	Value	Key 0	Value 0
	main_dsk2_switch_source	-3		

C.1.5 MEDIA PLAYERS

The shortcuts described in this section control settings and operations affecting TriCaster's various Media Players, such as DDR 1, DDR 2, GFX 1, and so on. Prefixes for use with these shortcuts are listed in Table 4.

Media Player Shortcuts	
Prefix	Details
ddr, ddr2	First and second DDR (media player)
stills	GFX 1 (media player)
titles	GFX2 (media player)
sound	Sound (media player)
focusedddr	Use the DDR that currently has focus.

TABLE 4

The prefixes listed in Table 3 above are valid targets for shortcuts listed in this section.

_playspeed(int)[state: (int)]

Description: Set the playback speed to **Value**, which represents a percentage (from 25-400) of normal speed.

Example:

Delay	Shortcut	Value	Key 0	Value 0
	ddr_playspeed	75		

_play(void)[state: (bool)]

Description: Play the current playlist item (if the player is not already playing).

Example:

Delay	Shortcut	Value	Key 0	Value 0
	ddr2_play			

_play_toggle(void)

Description: Toggle playback state (playing or stopped).

Example:

Delay	Shortcut	Value	Key 0	Value 0
	sound_play_toggle			

_stop(void)[state: (bool)]

Description: Stop playback.

Example:

Delay	Shortcut	Value	Key 0	Value 0
	ddr_stop			

_back(void)

Description: Select the previous item in the playlist, moving the playhead to its 'in point' (does not wrap around past first item).

Example:

Delay	Shortcut	Value	Key 0	Value 0
	ddr2_back			

_forward(void)

Description: Select the next playlist item, moving the playhead to its 'in point' (does not wrap around).

Example:

Delay	Shortcut	Value	Key 0	Value 0
	ddr2_forward			

_add_clips(string)

Description: Adds media files whose paths are defined in **Value** to the playlist; **Value** can define one file or multiple files separated by | (pipe symbol).

Example:

Delay	Shortcut	Value	Key 0
	ddr_add_clips	D:\show\intro.avi|D:\show\middle.avi	

_play_file(void)[path: (string)][index: (int)][tag: (string)]

Description: Plays item at the **Path** or with the **Tag** specified. The optional **Index** value controls which item plays when more than one identical item exists in the playlist (e.g., index="1" plays the 2nd clip; note that "-1" always plays the last item). The item will be added to the playlist from disk if necessary.

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	ddr2_play_file		path	D:\show\MyClip.avi	Index	-1

_select_file(path string, index int, tag string)

Description: Selects item at the **Path** or with the **Tag** specified. The **Index** value controls which item is selected when more than one identical item exists in the playlist (e.g., index="1" plays the 2nd clip; note that "-1" always selects the last item). The item will be added to the playlist from disk if necessary.

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	ddr2_play_file		Path	D:\show\MyClip.avi	Index	0

_move_playhead_to_clip(int)

Description: Moves the playhead to in point of an item at the playlist index (starts at 0) position supplied as **Value**.
Example:

Delay	Shortcut	Value
	ddr2_move_playhead_to_clip	3

_select_clip(int)

Description: Selects a playlist item represented by **Value** (indexing begins at 0).

Example:

Delay	Shortcut	Value
	ddr_select_clip	8

_select_clips(string)

Description: Selects multiple playlist items where **Value** supplies playlist indices formatted as n1|n2|n3, etc. (indexing begins at 0).

Example:

Delay	Shortcut	Value
	ddr_select_clips	8|9|11

_select_clips_by_tag(string)

Description: Selects one or more clips having the specified tag(s). Separate multiple tags using the pipe symbol (|).

Example:

Delay	Shortcut	Value
	ddr_select_clips_by _tag	Intro|Kiki1|RexTitle

_set_fade_transition_to_currently_selected_clips (void)

Description: Sets fade transition to all selected clips in the Media Player identified by the prefix.

Example:

Delay	Shortcut	Value
	ddr1__set_fade_transition_to_currently_selected_clips	

_select_preset(int)[state: (int)]

Description: Selects a preset specified by **Value** (starts at 0)

Example:

Delay	Shortcut	Value
	ddr2_select_preset	3

_copy_preset(int)

Description: Copies a preset specified by **Value** (starts at 0).

Example:

Delay	Shortcut	Value
	ddr_copy_preset	2

_paste_preset(int)

Description: Pastes a previously copied preset into the preset slot specified by **Value** (starts at 0).

Example:

Delay	Shortcut	Value
	ddr_paste_preset	5

_delete_preset(int)

Description: Deletes the preset specified by **Value** (starts at 0); leaves the first preset selected.

Example:

Delay	Shortcut	Value
	ddr_delete_preset	2

_remove_currently_selected_clips(void)

Description: Removes selected clips from the playlist.

Example:

Delay	Shortcut	Value
	ddr_ remove_currently_selected_clips	2

_relative_time(double)

Description: Scrubs the playhead from the current position by the **Value** supplied (in seconds). If the Media Player is in list playback mode, the playhead can move into the neighboring clip as required.

Example:

Delay	Shortcut	Value
	ddr_ relative_time	4.0

_relative_frame(int)

Description: Scrubs the playhead from the current position by the **Value** supplied (in frames). If the Media Player is in list playback mode, the playhead can move into the neighboring clip as required.

Example:

Delay	Shortcut	Value
	ddr_ relative_frame	20

_single_mode_toggle(void)[state: (bool)]

Description: Toggles single mode on or off; supply a Boolean **Value** to explicitly specify the result.

Example:

Delay	Shortcut	Value
	ddr2_single_mode_toggle	

_loop_mode_toggle(void)[state: (bool)]

Description: Toggle loop mode on or off; supply a Boolean Value to explicitly specify the result.

Example:

Delay	Shortcut	Value
	ddr2_loop_mode_toggle	

_autoplay_mode_toggle(void)[state: (bool)]

Description: Toggle autoplay mode on or off; supply a Boolean Value to explicitly specify the result.

Example:

Delay	Shortcut	Value
	ddr2_autoplay_mode_toggle	true

_next_preset(void)

Description: Selects the next preset for the Switcher, M/E or Media player designated by the prefix.

Example:

Delay	Shortcut	Value
	ddr_next_preset	

_previous_preset(void)

Description: Selects the previous preset for the Switcher, M/E or Media player designated by the prefix.

Example:

Delay	Shortcut	Value
	ddr_previous_preset	

_shuttle(int)

Description: Shuttle at speed **Value** specified as a percentage (from -1600 through 1600; 0 stops shuttling)

Example:

Delay	Shortcut	Value
	ddr2_shuttle	150

_scrub_to_time_from_beginning(double)

Description: Jumps the playhead by the **Value** specified (in seconds) from the item's **In** point (negative values jump back in time).

Example:

Delay	Shortcut	Value
	ddr_scrub_to_time_from_beginning	-3.0

_scrub_to_time_from_end(double)

Description: Jumps the playhead by the **Value** specified (in seconds) from the item's **Out** point (positive values jump back in time).

Example:

Delay	Shortcut	Value
	ddr2_scrub_to_time_from_end	5.0

_set_duration(double)

Description: Applies the duration supplied as Value to every selected playlist item.

Example:

Delay	Shortcut	Value	Key 0	Value 0
	ddr_set_duration	5		

_mark_in(double)

Description: Sets the **In** point to the time supplied as **Value** (with no **Value** supplied, updates the **In** point to the current playhead position).

Example:

Delay	Shortcut	Value
	ddr2_mark_in	5.0

_mark_out(double)

Description: Sets the **Out** point to the time supplied as **Value** (with no **Value** supplied, updates the **Out** point to the current playhead position).

Example:

Delay	Shortcut	Value
	ddr2_mark_out	5.0

_split(void)

Description: Splits the selected clip into two clips, dividing them at the playhead position (does not create a new file on disk).

Example:

Delay	Shortcut	Value
	ddr2_split	

_mark_in_reset(void)

Description: For the selected clip in the targeted Media Player, this shortcut will set the "**In**" point to 0. *For a full listing of valid Prefixes see Table 3.*

Example:

Delay	Shortcut	Value
	ddr_mark_in_reset	

_mark_out_reset(void)

Description: For the selected clip in the targeted Media Player this shortcut will set the "**Out**" point to the end of the clip. *For a full listing of valid Prefixes see Table 3.*

Example:

Delay	Shortcut	Value
	ddr_mark_out_reset	

_mark_in_out_clear(void)

Description: Clears the **In** and **Out** points for the selected playlist item

Example:

Delay	Shortcut	Value
	ddr2_mark_in_out_clear	

_copy_selected(void)

Description: Copy the selected playlist item into memory.

Example:

Delay	Shortcut	Value
	ddr_copy_selected	

_paste_selected(int)

Description: Paste item(s) in memory into a playlist. Specify insertion point using an index **Value** (-1 pastes at the start; -2 pastes at the end; null entry uses the current position).

Example:

Delay	Shortcut	Value
	ddr2_paste_selected	-1

_set_audio_level(clip_index int, clip_tag string, level double)

Description: Sets the volume level of a clip in the playlist designated by **clip_index** or **clip_tag**. The **level** value is in decibels.

Example:

_send_to_framebuffer(playlist_index int, clip_tag string, buffer_number int)

Description: Send a file to a specified framebuffer using either its **playlist index** (starts at 0) or **clip tag**.

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	ddr_send_to_framebuffer		playlist_index	2	buffer_number	3

_set_clip_tag(clip_index int, clip_tag string)

Description: Assigns a custom "tag" (_tag_string value) to identify a playlist item specified by the clip_index value.

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	ddr_set_clip_tag		clip_index	3	clip_tag	MyTag

_set_clip_alias(clip_index int, alias string)

Description: Set the display name (alias) of an item specified by its playlist index value (starts at 0).

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	ddr_set_clip_alias		clip_index	1	clip_alias	Bob

_set_clip_comment(clip_index int, comment string)

Description: Sets the comment for a playlist item specified by clip_index (starts at 0)

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	ddr2_set_clip_comment		clip_index	1	comment	Great shot!

_set_currentframe_as_thumbnail

Description: Sets the current frame of the selected clip it as the thumbnail.

Example:

Delay	Shortcut	Value	Key 0	Value 0
	ddr2_set_currentframe_as_thumbnail			

_add_to_playlist(filename string, current bool, playlists string, chop int, select bool, clip_tag string, comment string, index int, recording bool, recorder_index int)

Description: Adds a media file to one or more playlists in the targeted Media Player. Another instance of the item is added each time the shortcut is invoked. Key explanations follow below:

- **filename** – specify the full path of the file to be added
- **current** – adds to an item to the current playlist (which can be in addition to a list provided by the **playlists** key value). Takes a Boolean (**true** or **false**) value.
- **playlists** – allows for multiple playlists to be defined as targets for the file. The list is formatted as comma separated numbers (e.g., 1,3,5,9,12)
- **chop** – optional; uses the supplied **Value** as seconds to chop the In point of the added clip.
- **select** – optional; useful for DDRs with a playlist that is currently selected. This parameter defaults to **true**; a **false** value prevents the selection and play head change that normally follows adding items to a Media Player
- **clip_tag** – optional; tags the clip with the supplied string value.
- **comment** – defines the Comment metadata for the item added to the playlist
- **index** – position where the item is added; If index is negative, missing, or out of bounds, insertion occurs at the end of the playlist
- **recording** – clip is being actively recorded
- **recorder_index** – the recorder module index that is capturing this clip

Example:

Shortcut	Value	Key 0	Value 0	Key 1	Value 1	Key 2	Value 2
ddr2_add_to_playlist		filename	D:\myclip.avi	current	true	playlists	0,3,11

_remove_from_playlist(playlists string, index int)

Description: Removes one or more items from playlists in the targeted Media Player.

- **Playlists** – indexed starting at 0, with additional values separated by a comma.
- **Index** – note that a negative value removes the last clip. Positive values designate which files to remove. Values greater than the list size it will be ignored.

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	ddr2_remove_from_playlist		playlists	1,3,5	index	2

_add_to_playlist_finished(filename string)

Description: Finalizes adding to DDR from recorder and does the clean up.

Example:

Shortcut	Value	Key 0	Value 0	Key 1
ddr2_add_to_playlist_finished		filename	F:\Clip\cat.mov	

_publish_selected_clips

Description: Sends any selected clips to the Publish queue.

Example:

Delay	Shortcut	Value
	ddr2_publish_selected_clips	

_select_angle_index(int)

Description: Selects a matching clip from a different IsoCorder recorder source by recorder index (starts at 0).

Example:

Delay	Shortcut	Value
	ddr_select_angle_index	2

_select_angle_index_delta(int)

Description: Selects a matching clip from another IsoCorder recorder identified by an offset value.

Example:

Delay	Shortcut	Value
	ddr_select_angle_index_delta	1

_show_on_switcher(void)

Description: Start or ends a Media Player “Show On” operation, depending on the current state.

Example:

Delay	Shortcut	Value
	ddr1_ show_on_switcher	

_set_show_on_switcher_index(int)

Description: Selects the M/E or Switcher target for the Show On operation by index (the main Switcher is index 0, etc.)

Example:

Delay	Shortcut	Value
	ddr1_set_show_on_switcher_index	0

_set_show_on_switcher_in_transition_type(int)

Description: Sets the transition type for the Show On feature. **Value** is specified as 0 for background, 1 for cut, 2 for custom.

Example:

Delay	Shortcut	Value
	ddr1_ set_show_on_switcher_in_transition_type	2

set show_on_switcher_in_transition_index(int)

Description: Sets the custom in transition index for the Show On feature. **Value** is 0-8 index of the transition in popup transition bin.

Example:

Delay	Shortcut	Value
	ddr1_ set_show_on_switcher_in_transition_index	2

set show_on_switcher_out_transition_type(int)

Description: Sets the out transition type for the Show On feature. **Value** is specified as 0 for background, 1 for cut, 2 for custom.

Example:

Delay	Shortcut	Value
	ddr1_ set_show_on_switcher_out_transition_type	2

set show_on_switcher_out_transition_index(int)

Description: Sets the custom out transition index for the Show On feature. **Value** is 0-8 index of the transition in popup transition bin.

Example:

Delay	Shortcut	Value
	ddr1_show_on_switcher_transition_index	3

request_filebrowser_update(void)

Description: description needed

Example:

Delay	Shortcut	Value
	Request_filebrowser_update	

C.2 Inputs

The shortcuts described in this section refer to the inputs available in TriCaster's main *Switcher* and *M/Es* rows, allowing you to modify their names, provide them with descriptive comments, or to control their respective LiveMatte and Proc Amp features.

Prefixes for use with these shortcuts are listed in Table 5.

Shortcut Prefixes	Description
input 1 to input 8	The specified input
net	Net 1
net 2	Net 2
ddr	DDR 1
ddr2	DDR 2
stills	GFX 1
Titles	GFX 2
v1 – v8	M/E 1 to M/E 8, as specified
bfr1 to bfr15	The Buffer slot identified by the specified number

TABLE 5

Generally, the prefixes described in Table 2 above are valid targets for shortcuts listed in this section. Certain models may not support every prefix, however. For example, a shortcut whose purpose is to select the source for Input 8 will fail with on a TriCaster 460, which has just four inputs.

_long_name(string)[state_type(string)]

Description: The **Value** supplied provides the text for the Video field of the target. The state_type returns the previous value.

Example:

Delay	Shortcut	Value
	input3_long_name	Zeppelin Cam

_short_name(string)[state_type(string)]

Description: The **Value** supplied provides the text for the Switcher Button field of the target.

Example:

Delay	Shortcut	Value
	input3_short_name	Z Cam

_audio_name(string)[state_type(string)]

Description: The **Value** supplied provides the text for the Audio name for the target.

Example:

Delay	Shortcut	Value
	input3_audio_name	Zeppelin Cam

_comment(string)[state_type(string)]

Description: The **Value** supplied provides the Comment entry field for the target.

Example:

Delay	Shortcut	Value	Key 0	Value 0
	input3_comment	The view from above!		

_toggle_livematte(bool)

Description: Toggles the state of the LiveMatte feature for the target. Use a Boolean value **Value** ("0" or "1", "true" or "false") to set a specific state.

Example:

Delay	Shortcut	Value
	input3_toggle_livematte	1

_toggle_procamp(bool)

Description: Toggles the state of the Proc Amp feature for the target. Use a Boolean value **Value** ("0" or "1", "true" or "false") to set a specific state.

Example:

Delay	Shortcut	Value
	input3_toggle_procamp	false

C.2.1 INPUT SETTINGS

Most input specific parameters can be directly controlled, as discussed in this section.

[INPUT][SUFFIX] (AS SPECIFIED)

Description: For the specified **input**, enables, disables, or sets the parameter identified by the **suffix** according to the **Value** supplied.

- **targets:**
  - input1
  - input2
  - (etc.)
  - net
  - net2
  - ddr1
  - ddr2
  - gfx1
  - gfx2
  - bfr1
  - bfr2
  - (etc.)
  - v1
  - v2
  - (etc.)
- **Suffix**
  - `_procamp_is_enabled(bool)`
  - `_brightness(float)`
  - `_hue(float)`
  - `_contrast(float)`
  - `_saturation(float)`
  - `_red(float)`
  - `_green(float)`
  - `_blue(float)`
  - `_roffset(float)`
  - `_goffset(float)`
  - `_boffset(float)`
  - `_rgain(float)`
  - `_ggain(float)`
  - `_bgain(float)`

- _uoffset(float)
- _voffset(float)
- _matte_is_enabled(bool)
- _mattecolor_red(float)(bool)
- _mattecolor_green(float(bool))
- _mattecolor_blue(float)
- _is_lumakey(bool)
- _tolerance_c(float)
- _smoothness(float)
- _tolerance_y(float)
- _spill_tolerance(float)
- _spill_smoothness(float)

- (from _0_ to _7_)
    - _0_hotspot_is_enabled(bool)
    - _0_hotspot_y(float)
    - _0_hotspot_x(float)
    - _0_hotspot_size(float)

- _crop_enabled(bool)
- _crop_left(float)
- _crop_right(float)
- _crop_top(float)
- _crop_bottom(float)
- _crop_feathering(float)
- _crop_smooth_horizontal(float)
- _crop_smooth_vertical(float)

Examples:

Delay	Shortcut	Value
	ddr1_matte_is_enabled	true

Delay	Shortcut	Value
	ddr1__crop_left	.25

C.3 Titles

The shortcuts described in this section control settings and operations affecting TriCaster's various Media Players as well as Buffer content. Prefixes for use with these shortcuts are listed in Table 6.

Title Template Related Shortcuts	
Prefix	**Details**
ddr, ddr2	First and second DDR (media player)
stills	GFX 1 (media player)
titles	GFX2 (media player)
bfr1, bfr2,bfr3…	Buffer specified by number

TABLE 6

_title_begin_edit(playlist_index int, clip_tag string)

Description: Initiates a batch of edits to a single Title Page. This command requires a matching "_title_end_edit" shortcut, and cannot be nested (it will abort if attempted). Use one (only) of the following to identify the target Title page:

- **Playlist_index** – Specifies the Title Page to edit by its index in the playlist (starts at 0).
- **Clip_tag** – Specifies a Title Page by a previously assigned user tag.

Example:

Delay	Shortcut	Value	Key 0	Value 0
	gfx2_title_begin_edit		playlist_index	7

_title_end_edit(void)

Description: Ends a batch edit, writes to disk, and updates the output. Must follow a "_title_begin_edit", else it will be ignored.

Example:

Delay	Shortcut	Value	Key 0	Value 0
	gfx2_title_begin_edit	7	playlist_index	7
	(title edit shortcuts here ...)			
	gfx2_title_end_edit			

_title_set_line_property(playlist_index int, clip_tag string, line_number int, tag string, property string, value string)

Description: Sets the properties of a text line. Use either Playlist_index *or* Clip_tag to identify the target Title Page.

- **Playlist_index** – specifies the Title Page to edit by its index in the playlist (starts at 0; note, too, that Buffers are *always* index 0).
- **Clip_tag** – specifies a Title Page by a previously assigned user tag.
- **Line_number** – the index line in the page (starts at 0)
- **Tag** –specifies the line to edit on the Title Page by a previously assigned tag string.
- **Property** – defines which property of the line to edit, as listed below:
    - **Text** – value is a string comprising the text to be displayed
    - **FontFamily** – value is a string identifying the FontFamily (e.g., Arial) to be used
    - **FontWeight** – value can be "Bold" or "Normal"
    - **FontSize** – value is a double (e.g. , "36.75")
    - **FontStyle** – value can be "Italic" or "Normal"
    - **FontUnderline** – value is a Boolean ("True" or "False")
    - **AllCaps** – value is a Boolean ("True" or "False")

Example:

Shortcut	Value	Key 0	Value 0	Key 1	Value 1	Key 2	Value 2
gfx2_title_begin_edit		playlist_index	7	property	FontStyle	value	Italic

_title_set_image_property(playlist_index int, clip_tag string, image_number int, changeable_only bool, tag string, property string, value string)

Description: Sets the properties of an image in a Title Page. Use either Playlist_index *or* Clip_tag to identify the target Title Page.

- **Playlist_index** – specifies the Title Page to edit by its index in the playlist (starts at 0; note, too, that Buffers are *always* index 0).
- **Clip_tag** – specifies a Title Page by a previously assigned user tag.
- **Image_number** – the unique index number identifying the image on the Title Page.
- **Changeable_only** – defaults to "false", which means index includes all images, not just the changeable ones.
- **Tag** –specifies the line to edit on the Title Page by a previously assigned tag string.
- **Property**
  - **FillFileName** – file name including full path
  - **ImageStretch** – value can be "Fill" (Stretch)**,** "UniformToFill" (Fill Area), or "Uniform" (Show All Image)

Example:

Shortcut	Value	Key 0	Value 0	Key 1	Value 1	Key 2	Value 2
ddr2_title_set_image_property		playlist_index	0	tag	logo	FillFileName	D:\pic.jpg

_title_set_layer_enabled(playlist_index int, clip_tag string, preset int,layer_name string, value bool)

Description: Set the enabled state of a layer group. Primarily for animated titles.

- **Playlist_index** – specifies the Title Page by its index in the playlist (starts at 0), by tag (tag overrides index).
- **Clip_tag** – specifies a Title Page by a previously assigned user tag.
- **Preset** – the animation preset index you are targeting on this Title Page.
- **Layer_name** – The name of the layer group.
- **Value** – A Boolean value that answers the question "is it enabled or not?".

Example:

Shortcut	Value	Key 0	Value 0	Key 1	Value 1	Key 2	Value 2	Key 3	Value 3
bfr10_title_set_layer_enabled		playlist_index	3	clip_tag	logo	preset	3	layer_name	Layer2

Key 4	Value 4
value	true

_title_line_restore_defaults(playlist_index int, clip_tag string, line_number int,tag string)

Description: Restores the font size, font family, bold, italic, underline, and all caps to the original values. Only used by the Title Editor window.

- **Playlist_index** – specifies the Title Page by its index in the playlist (starts at 0), or by tag (tag overrides index).
- **Clip_tag** – specifies a Title Page by a previously assigned user tag.
- **Line_number** – the unique index number identifying the line on the Title Page.
- **Tag** – A previously tagged (named) line.

Example:

Shortcut	Value	Key 0	Value 0	Key 1	Value 1	Key 2	Value 2	Key 3	Value 3
bfr10_title_line_restore_defaults		playlist_index	3	clip_tag	logo	Line_number	3	tag	theLine 2

_title_image_restore_defaults(playlist_index int, clip_tag string, image_number int,tag string)

Description: Restores the fill mode and file name to the original values. Only used by the Title Editor window.

- **Playlist_index** – specifies the Title Page by its index in the playlist (starts at 0), by tag (tag overrides index).
- **Clip_tag** – specifies a Title Page by a previously assigned user tag.
- **Image_number** – the unique index number identifying the image on the Title Page.
- **Tag** – A previously tagged (named) line.

Example:

Shortcut	Value	Key 0	Value 0	Key 1	Value 1	Key 2	Value 2	Key 3	Value 3
bfr10_title_line_restore_defaults		playlist_index	3	clip_tag	logo	image_number	3	tag	Portrait 1

_title_save_png(playlist_index int, clip_tag string, destination string)

Description: Saves the current state of a Title Page to a PNG bitmap file. Setting the destination parameter requires a full file path. Use either Playlist_index *or* Clip_tag to identify the source Title Page.

Example:

Shortcut	Value	Key 0	Value 0	Key 1	Value 1
ddr2_title_save_png		playlist_index	3	destination	D:\mytitle.png

C.4 Positioner

The shortcuts described in this section control the Positioner options and settings for TriCaster's DSK and Key layers, as well as layers in M/Es when applicable.

Prefixes for use with these shortcuts are listed in Table 7.

Positioning Shortcuts	
Prefix	Details
main_dsk1-main_dsk4	Main switcher DSK specified by number (1-4)
virtualinputs_a - virtualinputs_d	Layer specified by letter (a-d) for delegated M/Es
virtualinputs_dsk1 – virtualinputs_dsk4	DSK specified by number (1-4) for delegated M/Es
v1_a, v1_b, v1_c, v1_d	Layer specified by letter (a-d) for M/E specified by number (V1-V8)
v1_dsk1, v1_dsk2, v1_dsk3, v1_dsk4	Key layer specified by number (1-4) for M/E specified by number (V1-V8)

TABLE 7

_position_x(double)

Description: Positions the targeted layer to the **Value** provided on the X axis. Valid values are between -500 and 500, where zero is the origin.

Example:

Delay	Shortcut	Value	Key 0	Value 0
	v1_a_position_x	120		

_position_y(double)

Description: Positions the targeted layer to the **Value** provided on the Y axis. Valid values are between -500 and 500, where zero is the origin.

Example:

Delay	Shortcut	Value	Key 0	Value 0
	v1_a_position_y	120		

_position_x_delta_value(double)

Description: Translates the targeted layer by the **Value** amount on the X axis.

Example:

Delay	Shortcut	Value	Key 0	Value 0
	v1_a_position_x_delta_value	10		

_position_y_delta_value(double)

Description: Translates the targeted layer by the **Value** amount on the Y axis. *Table 6.*

Example:

Delay	Shortcut	Value	Key 0	Value 0
	v1_a_position_y_delta_value	10		

_position_reset(void)

Description: Resets all Positioner values for the targeted layer to their origin.

Example:

Delay	Shortcut	Value	Key 0	Value 0
	v1_a_position_reset			

_scale_all(x double,y double, z double)

Description: Changes the scale on all axes to the specified values. Valid value ranges from: -500 to 500, where 100 is the origin.

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	v1_a_scale_all		x	25	y	25

_scale_x(double)

Description: Changes the scale on the X axis to the specified **Value**. Valid ranges are -500 to 500, with 100 as the origin.

Example:

Delay	Shortcut	Value
	v1_a_scale_x	25

_scale_y(double)

Description: Changes the scale on the Y axis to the **Value** specified. Valid ranges are -500 to 500, with 100 as the origin.

Example:

Delay	Shortcut	Value
	v1_a_scale_y	25

_scale_x_delta_value(double)

Description: Changes the scale X value by the amount specified by **Value**.

Example:

Delay	Shortcut	Value
	v1_a_scale_x_delta_value	10

_scale_y_delta_value(double)

Description: Changes the scale Y value by the amount specified by **Value**.

Example:

Delay	Shortcut	Value
	v1_a_scale_y_delta_value	10

_scale_delta_value(double)

Description: Changes the scale of both the X and Y values by the amount specified by **Value**.

Example:

Delay	Shortcut	Value
	v1_a_scale_delta_value	10

_scale_reset(void)

Description: Resets all scale values to their origin.

Example:

Delay	Shortcut	Value	Key 0
	v1_a_scale_reset		

_rotation_x(double)

Description: Sets the Rotate X setting to the supplied **Value**. Valid values range from -360 to 360.

Example:

Delay	Shortcut	Value	Key 0
	v1_a_rotation_x	45	

_rotation_y(double)

Description: Sets the Rotate Y setting to the supplied **Value**. Valid values range from -360 to 360.

Example:

Delay	Shortcut	Value	Key 0
	v1_a_rotation_y	45	

_rotation_z(double)

Description: Sets the Rotate Z setting to the supplied **Value**. Valid values range from -360 to 360.

Example:

Delay	Shortcut	Value
	v1_a_rotation_z	45

_rotation_x_delta_value(double)

Description: Changes the Rotate X setting by the supplied **Value**.

Example:

elay	Shortcut	Value
	v1_a_rotation_x_delta_value	-45

_rotation_y_delta_value(double)

Description: Changes the Rotate Y setting by the supplied **Value**.

Example:

Delay	Shortcut	Value
	v1_a_rotation_y_delta_value	-45

_rotation_z_delta_value(double)

Description: Changes the Rotate Z setting by the supplied **Value**.

Example:

Delay	Shortcut	Value
	v1_a_rotation_z_delta_value	-45

_rotation_reset(void)

Description: Resets all rotate values to their origin.

Example:

Delay	Shortcut	Value
	v1_a_rotation_reset	

_positioning_enable(bool)

Description: Sets the “Enabled” state of Positioning for the target layer to a value of either true or false.

Example:

Delay	Shortcut	Value
	v1_a_positioning_enable	true

_left_delta_value(double)

Description: Changes the Edges Left value by the supplied **Value**. Valid values are 0 to 100.

Example:

Delay	Shortcut	Value
	v1_a_left_delta_value	20

_right_delta_value(double)

Description: Changes the Edges Right value by the supplied **Value**. Valid values are 0 to 100.

Example:

Delay	Shortcut	Value
	v1_a_right_delta_value	20

_up_delta_value(double)

Description: Changes the Edges Top setting by the supplied **Value**.

Example:

Delay	Shortcut	Value
	v1_a_up_delta_value	20

_down_delta_value(double)

Description: Changes the Edges Bottom setting by the supplied **Value**.

Example:

Delay	Shortcut	Value
	v1_a_down_delta_value	20

_feather_delta_value(double)

Description: Changes the Edges Feather value by the supplied **Value**.

Example:

Delay	Shortcut	Value
	v1_a_feather_delta_value	5

_all_delta_value(double)

Description: Changes all Edges settings by the supplied **Value**.

Example:

Delay	Shortcut	Value
	v1_a_all_delta_value	5

_zindex_set_value(int)

Description: Changes the zindex or priority on DSKs.

Example:

Delay	Shortcut	Value
	v1_zindex_set_value	5

_crop_reset(void)

Description: Resets all Edges values to 0.

Example:

Delay	Shortcut	Value
	v1_a_crop_reset	

_lock_to_virtualset(bool)

Description: Changes the Lock to Virtual Set Value by (value).

Example:

Delay	Shortcut	Value
	v1_a_lock_to_virtualset	

_parallex_delta_value(double)

Description: Changes the Tracking parallex Value by (value).

Example:

Delay	Shortcut	Value
	v1_a_parallex_delta_value	

_zindex_set_value(int)

Description: Changes the zindex/priority on DSKs.

Example:

Delay	Shortcut	Value
	v1_a_zindex_set_value	

_crop_enable(bool)

Description: Sets the Enabled state of Edges for the target layer to value, true or false.

Example:

Delay	Shortcut	Value
	v1_a_crop_enable	true

_border_select_index(int)

Description: Select the Border preset for the target layer by index (starts at zero).

Example:

Delay	Shortcut	Value
	v1_a_border_select_index	1

_border_set(index int, path string)

Description: Select a file at **path** as the current selection for the **Border** preset identified by **index** (starts at zero).

Example:

Delay	Shortcut	Value	Key 0	Value 0
	v1_a_border_set_index	1	path	D:\MyFrame.PSD

_border_enable(bool)

Description: Sets the "Enabled" state of the Border feature for the target layer to a value of either true or false.

Example:

Delay	Shortcut	Value
	v1_a_border_enable	true

C.5 Color Group

The shortcuts described in this section control the Color Group settings for TriCaster's Switcher and M/E layers.

Prefixes for use with these shortcuts are listed in Table 8.

Positioning Shortcuts	
Prefix	Details
main_a; main_b	Main switcher DSK number (1-4)
v1_a, v1_b, v1_c, v1_d	Layer specified by letter (a-d) for M/E specified by number (V1-V8)

TABLE 8

_select_color_group(int)

Description: Assigns the target Switcher or M/E row to the color group specified by Value (begins at 0).

Example:

Delay	Shortcut	Value
	v1_a_select_color_group	0

_switch_color_group(int)

Description: get description and verify example

Example:

Delay	Shortcut	Value
	v1_a_select_color_group	0

_toggle_color_group(void)[state: (bool)]

Description: Toggles the color group assignment for the target Switcher or M/E row. The shortcut returns a Boolean state value.

Example:

elay	Shortcut	Value
	v1_a_toggle_color_group	

_clear_color_group(void)

Description: Removes the color group assignment for the target Switcher or M/E row.

Example:

Delay	Shortcut	Value
	v1_a_clear_color_group	

C.6 Undo/Redo

switcher_undo(bool)[state: (bool)]

Description: Reverts to the next previous preset stored for MAIN and MEs at prior Program row selection operation.

Example:

Delay	Shortcut	Value
	switcher_undo	

switcher_redo(bool)[state: (bool)]

Description: Reverts an Undo (restores preset stored for MAIN and MEs at next Program row selection operation).

Example:

Delay	Shortcut	Value
	switcher_redo	

C.7 PreViz

copy_main_to_previz(void)

Description: Copy Main to PREVIZ.

Example:

Delay	Shortcut	Value
	copy_main_to_previz	

copy_previz_to_main(void)

Description: Copy PREVIZ to Main.

Example:

Delay	Shortcut	Value
	copy_previz_to_main	

copy_me_to_previz(void)

Description: Copy M/E to PREVIZ.

Example:

Delay	Shortcut	Value
	copy_me_to_previz	

copy_previz_to_me(void)

Description: Copy PREVIZ to M/E.

Example:

Delay	Shortcut	Value
	copy_previz_to_me	

C.8 Effect View

fx_transition_main_toggle(void))[state: (bool)]

Description: Toggles the Effect View preview mode for the main Switcher. The shortcut returns a Boolean state value.

Example:

Delay	Shortcut	Value
	fx_transition_main_toggle	

fx_transition_main_take(void)

Description: Performs as Take of the Effect Transition preview for the main Switcher.

Example:

Delay	Shortcut	Value
	fx_transition_main_take	

fx_transition_main_auto(void)

Description: Autos the Effect Transition preview state onto output for the main Switcher.

Example:

Delay	Shortcut	Value
	fx_transition_main_auto	

fx_transition_me_toggle(void) [state: (bool)]

Description: Toggles the Effect View preview mode for the current (most recently selected) M/E. The shortcut returns a Boolean state value.

Example:

Delay	Shortcut	Value
	fx_transition_me_toggle	

fx_transition_me_take(void)

Description: Performs an M/E Take of the Effect Transition preview state for the current (last selected) M/E.

Example:

elay	Shortcut	Value
	fx_transition_me_take	

fx_transition_me_auto(void)

Description: Performs an M/E Auto of the Effect Transition preview state for the current (last selected) M/E.

Example:

Delay	Shortcut	Value
	fx_transition_me_auto	

C.9 RECORD

grab_still(string)

Description: Grabs still images; respects the settings in TriCaster's Grab dialog. The **string** value can be used to provide an optional full path to the jpg to be created. If the value is empty, the file is captured to the session Stills folder.

Example:

Delay	Shortcut	Value
	grab_still	D:\mypic.jpg

grab_filename(string)

Description: Sets the Grab base filename.

Example:

Delay	Shortcut	Value
	grab _filename	mygrabname

record_toggle(int)[state: (int)]

Description: Toggles recording on or off. Optionally enter a 1 or 0 **Value** to directly set the record state. (Setting **Value** to "3" enables recording and turns on bit 1 to signal blinking the CTRL key.)

Example:

Delay	Shortcut	Value
	record _toggle	1

record_start(void)

Description: Explicitly starts recording if not already underway.

Example:

Delay	Shortcut	Value

	record _start	

record_start_and_chop(void)

Description: Will start recording if not already recording.

Example:

Delay	Shortcut	Value
	record _start_and_chop	

record_stop(void)

Description: Explicitly stops recording.

Example:

Delay	Shortcut	Value
	record _stop	

record_reset(void)

Description: If recording is underway, discards the current recording and starts a new file; initiates recording if not already underway.

Example:

Delay	Shortcut	Value
	record _reset	

record_chop(int)

Description: If recording, performs a record chop; Starts recording if not already underway. The value (int) is for internal use.

Example:

Delay	Shortcut	Value
	record _chop	

record_setting(setting string, value string)

Description: uses name and value to set settings in record panel ex. setting:row_1_ischecked value:true or setting:row_1_source value: input1.

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1

	record _setting		Setting	row_1_ischecked	value	True

record_chop_source(int)

Description: The int value is the index of the recorder module.

Example:

Delay	Shortcut	Value
	record _chop_source	

record_filename(string)

Description: Sets the Record base filename.

Example:

elay	Shortcut	Value
	record _filename	myclipname

record_replay_panel (void)

Description: Triggers a *Replay Pad* operation on the recorder specified by **index**. The **replay** value (true or false) controls the operation of the associated *Instant Replay* feature.

Example:

elay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	record _replay_panel		index	0	replay	true

C.10 Audio Mixer

The shortcuts described in this section control the settings and operations of TriCaster's Audio Mixer.

Shortcut details are grouped in several sub-sections, each with its own table of supported prefixes.

C.10.1 INPUTS AND OUTPUTS

Audiomixer Shortcuts	
Prefix	Details
input1 – 8	Input specified by number
ddr, ddr2	DDR 1, DDR 2
stills	GFX 1 (media player)
titles	GFX2 (media player)
net, net2	Net input 1, Net input 2
music	Sound (media player)
effects	Animation Store transition sound level

outputaux, outputaux2	Aux 1 output, Aux 2 output
outputstream	Streaming output
outputrecord	Record level
master, master2	Master 1, Master 2

TABLE 9

_volume(double)[state: (double)]

Description: Set the volume level of the target to the supplied Value in dB. **State** returns the current volume.

Example:

Delay	Shortcut	Value
	ddr_volume	-20

_volume2(double)[state: (double)]

Description: Sets the volume level of the of the target's second slider to the supplied Value in dB. **State** returns the current volume.

Example:

Delay	Shortcut	Value
	ddr_volume2	-20

_pan(pan double, channel int)[state: (pan double, channel int)]

Description: Sets the pan setting for the target to **Value**, in decibels (dB). **State** returns the current pan setting.

- **Pan** – the **Value** (dB) to set.
- **Channel** – represents channels A-D using integers from 0-3.

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	ddr_pan		pan	10	channel	2

_adjust_pan(pan double, channel int)[state: (pan double, channel int)]

Description: Modifies the pan setting for the target by **Value**, in decibels (dB). **State** returns the current pan setting.

- **Pan** – the **Value** (dB) by which the Pan setting will be modified.
- **Channel** – represents channels A-D using integers from 0-3.

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	ddr_adjust_pan		pan	-10	channel	0

_adjust_volume(double)

Description: Modifies the volume of the target by the supplied **Value** in decibels (dB).

Example:

Delay	Shortcut	Value
	ddr_adjust_volume	16

_adjust_volume2(double)

Description: Modifies the volume of the target's second slider by the supplied **Value** in decibels (dB).

Example:

Delay	Shortcut	Value
	ddr_adjust_volume2	16

_mute(bool)[state: (bool)]

Description: Turns Mute on or off for the target.). **State** returns the current Mute setting.

Example:

Delay	Shortcut	Value
	ddr_mute	0

_mute2(bool)[state: (bool)]

Description: Turns Mute on or off for the target's second slider.). **State** returns the current Mute setting.

Example:

Delay	Shortcut	Value
	ddr_mute2	0

_mute_toggle(void)[state: (bool)]

Description: Toggles Mute on or off for the target. **State** returns the current Mute setting.

Example:

Delay	Shortcut	Value
	ddr_mute_toggle	

_mute2_toggle(void)[state: (bool)]

Description: Toggles Mute on or off for the target's second slider. **State** returns the current Mute setting.

Example:

Delay	Shortcut	Value
	ddr_mute_toggle2	

_mono(bool)[state: (bool)]

Description: Modifies the Pan settings for the target to enable or disable a Mono state. **State** returns the current setting.

Example:

Delay	Shortcut	Value
	ddr2_mono	

_solo(bool)[state: (bool)]

Description: Toggles Solo on or off for the target. **State** returns the current Solo setting.

Example:

Delay	Shortcut	Value
	ddr2_solo	

_solo_toggle(bool)[state: (bool)]

Description: Turns target Solo On/Off.

Example:

Delay	Shortcut	Value
	ddr2_solo_toggle	

_enable_equalizer(bool)[state: (bool)]

Description: Enables or disables the Equalizer for the target based on supplied Boolean **Value** (0 or 1, true or false). **State** returns the current Equalizer state.

Example:

Delay	Shortcut	Value
	ddr2_enable_equalizer	1

_enable_compressor(bool)[state: (bool)]

Description: Enables or disables the Compressor/Limiter for the target based on supplied Boolean **Value** (0 or 1, true or false). **State** returns the current Compressor/Limiter state.

Example:

Delay	Shortcut	Value

	ddr2_enable_compressor	1

_reset_equalizer(void)[state: (void)]

Description: Reset the target's Equalizer to its default values.

Example:

Delay	Shortcut	Value
	ddr2_reset_equalizer	

_reset_compressor(void)[state: (void)]

Description: Reset the target's Compressor/Limiter to its default values.

Example:

Delay	Shortcut	Value
	ddr2_reset_compressor	

_eq_slider_0(double)[state: (double)]

Description: Set Equalizer slider 0 to the **Value** supplied. **State** returns the current slider state.

Example:

Delay	Shortcut	Value
	ddr2_eq_slider_0	-4

_eq_slider_1(double)[state: (double)]

Description: Set Equalizer slider 1 to the **Value** supplied. **State** returns the current slider state.

Example:

Delay	Shortcut	Value
	ddr2_eq_slider_1	-4

_eq_slider_2(double)[state: (double)]

Description: Set Equalizer slider 2 to the **Value** supplied. **State** returns the current slider state.

Example:

Delay	Shortcut	Value
	ddr2_eq_slider_2	-4

_eq_slider_3(double)[state: (double)]

Description: Set Equalizer slider 3 to the **Value** supplied. **State** returns the current slider state.

Example:

Delay	Shortcut	Value
	ddr2_eq_slider_3	-4

_eq_slider_4(double)[state: (double)]

Description: Set Equalizer slider 4 to the **Value** supplied. **State** returns the current slider state.

Example:

Delay	Shortcut	Value
	ddr2_eq_slider_4	-4

_eq_slider_5(double)[state: (double)]

Description: Set Equalizer slider 5 to the **Value** supplied. **State** returns the current slider state.

Example:

Delay	Shortcut	Value
	ddr2_eq_slider_5	-4

_eq_slider_6(double)[state: (double)]

Description: Set Equalizer slider 6 to the **Value** supplied. **State** returns the current slider state.

Example:

Delay	Shortcut	Value
	ddr2_eq_slider_6	-4

_gain_0(double)[state: (double)]

Description: Set Gain 0 to the **Value** supplied. **State** returns the current state.

Example:

Delay	Shortcut	Value
	ddr2_gain_0	-4

_gain_1(double)[state: (double)]

Description: Set Gain 1 to the **Value** supplied. **State** returns the current state.

Example:

Delay	Shortcut	Value
	ddr2_gain_1	-4

_gain_2(double)[state: (double)]

Description: Set Gain 2 to the **Value** supplied. **State** returns the current state.

Example:

Delay	Shortcut	Value
	ddr2_gain_2	-4

_gain_3(double)[state: (double)]

Description: Set Gain 3 to the **Value** supplied. **State** returns the current state.

Example:

Delay	Shortcut	Value
	ddr2_gain_3	-4

_gain_4(double)[state: (double)]

Description: Set Gain 4 to the **Value** supplied. **State** returns the current state.

Example:

Delay	Shortcut	Value
	ddr2_gain_4	-4

_gain_5(double)[state: (double)]

Description: Set Gain 5 to the **Value** supplied. **State** returns the current state.

Example:

Delay	Shortcut	Value
	ddr2_gain_5	-4

_gain_6(double)[state: (double)]

Description: Set Gain 6 to the **Value** supplied. **State** returns the current state.

Example:

Delay	Shortcut	Value
	ddr2_gain_6	-4

_gain_7(double)[state: (double)]

Description: Set Gain 7 to the **Value** supplied. **State** returns the current state.

Example:

Delay	Shortcut	Value
	ddr2_gain_7	-4

_routing_mask_0(ulong)[state: (ulong)]

Description: Set the 1st routing mask to Value. Routing mask is a 64-bit unsigned integer. Little-endian Indexing each bit, produces a Boolean representing the routing of output (i/8) to input (i%8). Routing Mask is a 64 bit unsigned integer. Little-endian indexing each bit, i, produces a Boolean representing the routing of output (i/8) to input (i%8). Prefix determines which input is being modified, masks 0-3 will apply the value to Master, Aux1, Aux2, or Aux3 accordingly.

Example:

Delay	Shortcut	Value
	Input1_routing_mask_0	9241421705770172929

_routing_mask_1(ulong)[state: (ulong)]

Description: Set the 2nd routing mask to Value. Routing mask is a 64-bit unsigned integer. Little-endian Indexing each bit, produces a Boolean representing the routing of output (i/8) to input (i%8). Routing Mask is a 64 bit unsigned integer. Little-endian indexing each bit, i, produces a Boolean representing the routing of output (i/8) to input (i%8). Prefix determines which input is being modified, masks 0-3 will apply the value to Master, Aux1, Aux2, or Aux3 accordingly.

Example:

Delay	Shortcut	Value
	Input1_routing_mask_1	9241421705770172929

_routing_mask_2(ulong)[state: (ulong)]

Description: Set the 3rd routing mask to Value. Routing mask is a 64-bit unsigned integer. Little-endian Indexing each bit, produces a Boolean representing the routing of output (i/8) to input (i%8). Routing Mask is a 64 bit unsigned integer. Little-endian indexing each bit, i, produces a Boolean representing the routing of output (i/8) to input (i%8). Prefix determines which input is being modified, masks 0-3 will apply the value to Master, Aux1, Aux2, or Aux3 accordingly.

Example:

Delay	Shortcut	Value
	Input1_routing_mask_2	9241421705770172929

_routing_mask_3(ulong)[state: (ulong)]

Description: Set the 4th routing mask to Value. Routing mask is a 64-bit unsigned integer. Little-endian Indexing each bit, produces a Boolean representing the routing of output (i/8) to input (i%8). Routing Mask is a 64 bit unsigned integer. Little-endian indexing each bit, i, produces a Boolean representing the routing of output (i/8) to input (i%8). Prefix determines which input is being modified, masks 0-3 will apply the value to Master, Aux1, Aux2, or Aux3 accordingly.

Example:

Delay	Shortcut	Value
	Input1_routing_mask_3	9241421705770172929

_cl_knob_0(double)[state: (double)]

Description: Set Compressor knob 0 to the supplied **Value** supplied. **State** returns the current slider state.

Example:

Delay	Shortcut	Value
	ddr_cl_knob_0	15

_cl_knob_1(double)[state: (double)]

Description: Set Compressor knob 1 to the supplied **Value** supplied. **State** returns the current slider state.

Example:

Delay	Shortcut	Value
	ddr_cl_knob_1	15

_cl_knob_2(double) [state: (double)]

Description: Set Compressor knob 2 to the supplied **Value** supplied. **State** returns the current slider state.

Example:

Delay	Shortcut	Value
	ddr_cl_knob_2	15

_cl_knob_3(double)[state: (double)]

Description: Set Compressor knob 31 to the supplied **Value** supplied. **State** returns the current slider state.

Example:

Delay	Shortcut	Value
	ddr_cl_knob_3	15

_cl_knob_4(double)[state: (double)]

Description: Set Compressor knob 4 to the supplied **Value** supplied. **State** returns the current slider state.

Example:

Delay	Shortcut	Value
	ddr_cl_knob_4	15

_balance(double)[state: (double)]

DEPRECATED. Retained for 3rd party backward compatibility. Now controls Pan on Channels 1 and 2.

_mic1_balance(double)[state: (double)]

DEPRECATED. Retained for 3rd party backward compatibility. Now controls Pan on Channel 1.

_mic2_balance(double)[state: (double)]

DEPRECATED. Retained for 3rd party backward compatibility. Now controls Pan on Channel 2.

_mic1_adjust_balance(double)[state: (double)]

DEPRECATED. Retained for 3rd party backward compatibility. Now controls Pan on Channel 1.

_mic2_adjust_balance(double)[state: (double)]

DEPRECATED. Retained for 3rd party backward compatibility. Now controls Pan on Channel 2.

_enable_noisegate(bool)[state:(bool)]

Set enable noisegate to true or false.

_noisegate_level(double)[state:(double)]

Set noisegate level to {value}.

C.10.2 INPUT SPECIFIC

TABLE 10

Audiomixer Shortcuts Group 2	
Prefix	Details
input1 – 8	Input specified by number
net, net2	Net input 1, Net input 2
ddr, ddr2	Input specified by number
music	Sound (media player)
effects	Animation Store transition sound level

_mic1_adjust_volume(double)

DEPRECATED. Retained for 3rd party backward compatibility.

_mic2_adjust_volume(double)

DEPRECATED. Retained for 3rd party backward compatibility.

_mic1_volume(double)[state: (double)]

DEPRECATED. Retained for 3rd party backward compatibility.

_mic2_volume(double)[state: (double)]

DEPRECATED. Retained for 3rd party backward compatibility.

_mic1_mute(bool)[state: (bool)]

DEPRECATED. Retained for 3rd party backward compatibility.

_mic2_mute(bool)[state: (bool)]

DEPRECATED. Retained for 3rd party backward compatibility.

_mic1_mute_toggle(void)[state: (void)]

DEPRECATED. Retained for 3rd party backward compatibility.

_mic2_mute_toggle(void)[state: (void)]

DEPRECATED. Retained for 3rd party backward compatibility.

_mic1_adjust_trim(double)

Description: Adjusts the target input’s Mic1 trim by the supplied **Value** in decibels.

Example:

Delay	Shortcut	Value
	input1_mic1_adjust_trim	5

_mic2_adjust_trim(double)

Description: Adjusts the target input's Mic2 trim by the supplied **Value** in decibels.

Example:

Delay	Shortcut	Value
	input1_mic2_adjust_trim	5

_mic1_trim(double)[state: (double)]

Description: Sets the trim of the target input's Mic1 to the supplied **Value** in decibels. **State** returns the current trim setting.

Example:

Delay	Shortcut	Value
	input1_mic1 _trim	15

_mic2_trim(double)[state: (double)]

Description: Sets the trim of the target input's Mic2 to the supplied **Value** in decibels. **State** returns the current trim setting.

Example:

Delay	Shortcut	Value
	input1_mic2 _trim	15

_audio_latency(double)[state: (double)]

Description: Sets Audio Delay for the target to supplied **Value** in 'frames'. **State** returns the current setting.

Example:

Delay	Shortcut	Value
	Input4_audio_latency	2

_input_type(string)[state: (string)]

Description: Set the connection type for the target input. **State** returns the current setting. Valid **Values** are:

- mic

- mic_single
- mic_and_phantom
- line
- aes3
- sdi_embedded
- line_quad

Example:

elay	Shortcut	Value
	input4_input_type	Mic

_follow(bool)[state: (bool)]

Description: Turns Follow for the target input on or off based on the Boolean **Value** (0 or 1) supplied. **State** returns the current setting.

Example:

Delay	Shortcut	Value
	ddr_follow	1

_follow_sources(string)[state: (string)]

Description: Sets sources to Follow one or more switcher inputs. Use the pipe character "|" as delimiters. **State** returns the current settings.

Example:

Delay	Shortcut	Value
	ddr_follow_sources	3|8|7|11

_talk(bool)[state: (bool)]

Description: Turns the "Talk Over" feature on or off for a Mic input based on the Boolean **Value** supplied (0 or 1). **State** returns the current settings.

Example:

Delay	Shortcut	Value
	input1_talk	

_talk_toggle(void)[state: (bool)]

Description: Turns input Talk On/Off.

Example:

Delay	Shortcut	Value
	input1_talk_toggle	

_color_group(int)[state: (int)

DEPRECATED. Retained for 3rd party backward compatibility.

C.10.3 ROUTING

AUDIO_ROUTING_DATA_[TARGET](VOID), PROPERTY_NAME(STRING), PROPERTY_VALUE(STRING)

Description: For the specified **target**, enables or disables the audio routing connection identified by the **property_name** value according to the boolean **property_value** supplied (e.g., the property_value “AbMain1” identifies a connection between the target’s first two channels, a and b, and “Master 1”).

- **targets:**
  - input1
  - input2, etc.
  - ddr
  - ddr2
  - music
  - effects
  - net
  - net2
  - stream
  - master
  - aux
- **property_name**
  - AbMain1
  - CdMain1
  - AbMain2
  - CdMain2
  - AbAux1
  - CdAux1
  - AbAux2
  - CdAux2
- **property_value** – Boolean (True, False)

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	audio_routing_data_input2		property name	AbAux1	property_value	true

```
<shortcut name=” audio_routing_data_input2”, property name=”AbAux1”/>
```

C.10.4 GENERAL

This section includes shortcuts that target the Audio Mixer or aspects of it but do not require prefixes.

_trim(double)[state: (double)]

Description: Set trim of the hardware input mic to (value) dB. State contains current trim.

audiomixer_solo(string)[state: (string)]

Description: Controls the Solo setting of multiple Audio Mixer inputs (outputs are not supported for multi-selection. **State** returns a corresponding string.

Example:

Delay	Shortcut	Value	Key 0	Value 0
	audiomixer_solo	input4|net2|ddr|input2		

audiomixer_save_to_emem(int)

Description: Store audiomixer state into mem preset at index **value**.

Example:

Delay	Shortcut	Value	Key 0	Value 0
	audiomixer_save_to_emem	4		

audiomixer_restore_default_emem(int)

Description: Delete audiomixer preset at index **value**.

Example:

Delay	Shortcut	Value	Key 0	Value 0
	audiomixer_restore_default_emem	0		

audiomixer_load_from_emem(int)

Description: Select audiomixer mem preset at index **value**.

Example:

Delay	Shortcut	Value	Key 0	Value 0
	audiomixer_load_from_emem	2		

phones_adjust_volume (double)[state: (double)]

Description: Changes the current headphone volume level by the supplied **Value** in decibels. **State** returns the current setting.

Example:

Delay	Shortcut	Value	Key 0	Value 0
	phones_adjust_volume	-2		

phones_volume(double)[state: (double)]

Description: Sets the headphone volume level to the supplied **Value** in decibels. **State** returns the current setting.

Example:

elay	Shortcut	Value	Key 0	Value 0
	phones_ volume	-6		

outputrecord_adjust_volume(double)[state: (void)]

Description: Modifies the Record volume level by the supplied **Value** in decibels. **State** returns the current setting.

Example:

elay	Shortcut	Value	Key 0	Value 0
	record_adjust_ volume	-3		

outputrecord_volume(double)[state: (double)]

Description: Sets the Record volume level to the supplied **Value** in decibels. **State** returns the current setting.

Example:

elay	Shortcut	Value	Key 0	Value 0
	record_ volume	-6		

C.11 OTHER

virtualinputs_follow_preview(bool)[state: (bool)]

Description: Enables or disables the "Virtual Inputs Delegate Follows Preview" option according to the supplied Boolean **Value**. **State** returns the current setting.

Example:

Delay	Shortcut	Value	Key 0	Value 0
	virtualinputs_follow_preview	1		

streaming_toggle(int)

Description: Toggles recording on/off; supply a Boolean **Value** (0 or 1) to specify a state.

Example:

elay	Shortcut	Value	Key 0	Value 0
	virtualinputs_follow_preview	1		

switcher_begin_changes(void)[state: (bool)]

Description: This shortcut is used with "**switcher_finish_and_send_changes**" to treat a sequence of switcher shortcuts as a single switcher message, intercepting all messages to the switcher until "switcher_finish_and_send_changes" is sent.

switcher_finish_and_send_changes(void)

Description: This shortcut is used with "**switcher_begin _changes**" to send a preceding sequence of switcher shortcuts as a single switcher message.

display_version(void)

Description: Use to display your version of TriCaster in the Dashboard's Error message display.

Example:

Delay	Shortcut	Value	Key 0	Value 0
	display_version			

set_router_mapping (void) [router_input (int), system_input (int)]

Description: Will map a router input to a TriCaster or 3Play input. Can optionally use the following DEPRECATED parameters (retained for LEGACY SUPPORT ONLY):

- **Input** = int
- **Output** = int
- **Tcinput** = int

Example:

Shortcut	Value	Key 0	Value 0	Key 1	Value 1
set_router_mapping		router_input	3	system_input	5

add_ip_camera(void) [camera_name (string), address (string)]

Description: Adds a new IP camera using supplied **camera_name** and **address**.

Example:

hortcut	Value	Key 0	Value 0	Key 1	Value 1
set_router_mapping		router_input	3	system_input	5

net_source(string)

Description: Sets the source for Net Input 1 to the **Value** entered as a string.

Example:

Delay	Shortcut	Value	Key 0
	net_source	My3Play(A)	

net2_source(string)

Description: Sets the source for Net Input 2 to the **Value** entered as a string.

Example:

Delay	Shortcut	Value	Key 0
	net2_source	My3Play(B)	

set_ip_source(index int,source_name string)

Description: Select a new IP source for the switcher IP source input button.

- Index – The index value of the input being assigned a new IP source. Index start at 0.
- Source_name-the source you are assigning to the input.

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	set_ip_source		index	0	source_name	IN 2

force_low_bandwidth_input(index int, value bool)

Description: ??

Example:

Delay	Shortcut	Value	Key 0
	force_low_bandwidth_input		

set_autodetect_psf_input(hw_index int, value bool)

Description: get details

Example:

Delay	Shortcut	Value	Key 0
	Set_autodetect_psf_input	My3Play(B)	

datalink_set(void) [key(string), value (string)]

Description: Specifies a DataLink **key** or creates a new one, and sets its **value**.

- **key** – the name of the Datalink key to set.
- **value** – the value to be assigned to the key specified.

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	datalink_set		key	player_12	value	Troy Willard

send_to_service(void) [destination (string), message (string)]

Description: Sends a message to a registered service (this is intended to allow external devices to be controlled by the TriCaster scheduling, macros, etc.)

play_macro_byname(string)

Description: Plays a macro having the name provided as **Value**.

Example:

Delay	Shortcut	Value	Key 0	Value 0
	play_macro_byname	My_Macro		

stop_all_macros(void)

Description: Stops all macros (including multistep macros).

Example:

Delay	Shortcut	Value	Key 0	Value 0
	stop_all_macros			

stop_macro_byname(string)

Description: Stops all macros with a given name.

Example:

Delay	Shortcut	Value	Key 0	Value 0
	stop_macro_byname	My_Macro		

stop_macro_byid(string)

Description: Stops a single macro with it unique id.

Example:

Delay	Shortcut	Value	Key 0	Value 0
	stop_macro_byname	My_Macro		

continue_pausedmacro(void)

Description: Continues a multistep macro, from the last wait command.

Example:

Delay	Shortcut		Key 0	Value 0
	continue_pausedmacro			

http_request(void)

Description: Should be used for sending commands to devices such as PTZ cameras within a macro. Will only trigger an HTTP GET request, response is unavailable.

Example:

Delay	Shortcut		Key 0	Value 0
	http_request	http://localhost:5952/v1/shortcut?name=man_auto		

The following group of “show_...” shortcuts are documented for 3rd-party purposes and are not recorded.

show_ddr1_tab(void)

Example:

Delay	Shortcut	Value	Key 0	Value 0
	show_ddr1_tab			

show_ddr2_tab(void)

Example:

Delay	Shortcut	Value	Key 0	Value 0
	show_ddr2_tab			

show_ddr3_tab(void)

Example:

Delay	Shortcut	Value	Key 0	Value 0
	show_ddr3_tab			

show_ddr4_tab(void)

Example:

Delay	Shortcut	Value	Key 0	Value 0
	show_ddr4_tab			

show_gfx1_tab(void)

Example:

Delay	Shortcut	Value	Key 0	Value 0
	show_gfx1_tab			

show_gfx2_tab(void)

Example:

Delay	Shortcut	Value	Key 0	Value 0
	show_gfx2_tab			

show_sound_tab(void)

Example:

Delay	Shortcut	Value	Key 0	Value 0
	show_sound_tab			

show_ptz_tab(void)

Example:

Delay	Shortcut	Value	Key 0	Value 0
	show_ptz_tab			

show_buffers_tab(void)

Example:

Delay	Shortcut	Value	Key 0	Value 0
	show_buffers_tab			

show_audiomixer_tab(void)

Example:

Delay	Shortcut	Value	Key 0	Value 0
	show_audiomixer_tab			

show_v1_tab(void)

Example:

Delay	Shortcut	Value	Key 0	Value 0
	show_v1_tab			

show_v2_tab(void)

Example:

Delay	Shortcut	Value	Key 0	Value 0
	show_v2_tab			

show_v3_tab(void)

Example:

Delay	Shortcut	Value	Key 0	Value 0
	show_v3_tab			

show_v4_tab(void)

Example:

Delay	Shortcut	Value	Key 0	Value 0
	show_v4_tab			

show_v5_tab(void)

Example:

Delay	Shortcut	Value	Key 0	Value 0
	show_v5_tab			

show_v6_tab(void)

Example:

Delay	Shortcut	Value	Key 0	Value 0
	show_v6_tab			

show_v7_tab(void)

Example:

Delay	Shortcut	Value	Key 0	Value 0
	show_v7_tab			

show_v8_tab(void)

Example:

Delay	Shortcut	Value	Key 0	Value 0
	show_v8_tab			

show_previz_tab(void)

Example:

Delay	Shortcut	Value	Key 0	Value 0
	show_previz_tab			

show_external_audiomixer(void)

Example:

Delay	Shortcut	Value	Key 0	Value 0
	show_external_audiomixer			

show_internal_audiomixer(void)

Example:

Delay	Shortcut	Value	Key 0	Value 0
	show_internal_audiomixer			

APPENDIX D. 3PLAY COMMANDS

D.1 Control surface and UI

record(bool)[state: (bool)]

Description: Start recording if no value is provided. A value of true or 1 will set the recording to start, and a value of false or 0 will set the recording to stop.

Example:

Delay	Shortcut	Value
	record	

stop-record(bool)

Description: Stop the recording.

Example:

Delay	Shortcut	Value
	stop-record	

record_blink(bool)[state: (bool)]

Description: Blinks the shift key when recording and pressed. A value of true will signal the shift key ON and a value of false will signal the shift key OFF.

Example:

Delay	Shortcut	Value
	record_blink	true
	record	

output(int)[state: (int)]

Description: Use to set the output being controlled. 0 = output A and 1 = output B.

Example:

Delay	Shortcut	Value
	output	1

output-sync(bool)

Description: Toggle linking the transport controls for all outputs. For example if linked when the operator plays output A then output B also plays.

Example:

Delay	Shortcut	Value
	output-sync	

mode(int)[state: (int)]

Description: Sets the mode for the controlled output. Mode Values: 0 = clips, 1 = playlist, 2 = live, and 3 = delayed.

Example:

Delay	Shortcut	Value
	mode	1

mark-in(bool)

Description: Mark an inpoint for what is currently live or delayed.

Example:

Delay	Shortcut	Value
	mark-in	

mark-in-update

Description: Update the inpoint of the clip/playlist item currently on the output being controlled.

Example:

Delay	Shortcut	Value
	mark-in-update	

mark-out

Description: Mark an outpoint for what is currently live or delayed.

Example:

Delay	Shortcut	Value
	mark-out	

mark-out-update

Description: Update the outpoint of the clip/playlist item currently on the output being controlled.

Example:

Delay	Shortcut	Value
	mark-out-update	

stop(bool)

Description: Stop playback on the controlled output.

Example:

Delay	Shortcut	Value
	stop	

play(bool)

Description: Start playback on the controlled output.

Example:

Delay	Shortcut	Value
	play	

play-speed (int)[state: (int)]

Description: Ramp up/Ramp down the playback speed for the controlled output. Values represent a percentage of play-speed with a value of 100 representing normal play-speed. If stopped and ramped up from zero the output will automatically start playback.

Example:

Delay	Shortcut	Value
	play-speed	150

jog (int)

Description: Jog the controlled output.

Example:

Delay	Shortcut	Value
	jog	

local-jog (int)

Description: Jog around in the clip/playlist item currently on the output being controlled.

Example:

Delay	Shortcut	Value
	local-jog	

fast-jog (bool)

Description: Toggle fast jogging mode on the control surface.

Example:

Delay	Shortcut	Value
	fast-jog	

set_airsend_autoplay (bool)[state: (bool)]

Description: Turn autoplay on/off for outputs connected to a downstream switcher's network input.

Example:

Delay	Shortcut	Value
	set_airsend_autoplay	1

play-clip-with-id(id string, camera int)

Description: Play clip "Event ID" and specify the camera angle.

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	play-clip-with-id		id	0-13	camera	3

add-to-list

Description: Add the currently selected clips to the current playlist tab.

Example:

Delay	Shortcut	Value
	add-to-list	

esc

Description: Escape out of the current editing operation.

Example:

Delay	Shortcut	Value
	esc	

remove

Description: When the controlled output is in Clip List mode, deletes the selected clips. In play list mode it deletes the selected playlist item.

Example:

Delay	Shortcut	Value
	remove	

enter

Description: Commit the current editing operation.

Example:

Delay	Shortcut	Value
	enter	

enter-shft

Description: Commit the current editing operation. If renaming a clip camera angle, it renmes all clip camera angles.

Example:

Delay	Shortcut	Value
	enter-shift	

loop (bool)[state: (bool)]

Description: Changes playback looping for the controlled output. A value of 0 will set loop to OFF and a value of 1 will set it ON. If no shortcut value is specified then looping is toggled.

Example:

Delay	Shortcut	Value
	loop	

edit-tbar (int)

Description: Adjusts the current spreadsheet cell's value by Value percentage.

Example:

Delay	Shortcut	Value
	edit-tbar	

edit-speed (int)

Description: Set the preset speed for the currently selected Playlist item to Value. "Speed" cell for Playlist item must be selected.

Example:

Delay	Shortcut	Value
	edit-speed	25

set-bookmark

Description: Creates a bookmark at the current live time being recorded.

Example:

Delay	Shortcut	Value
	set-bookmark	

goto-prevbookmark

Description: Traverse the 10 bookmarks starting with the most recently created one.

Example:

Delay	Shortcut	Value
	goto-prevbookmark	

tagging (bool)[state: (bool)]

Description: Set the tagging mode. When Value is true the control surface numpad is used to apply tags, when false the numpad numbers are entered into the UTEB. If no Value then the tagging mode is toggled.

Example:

Delay	Shortcut	Value
	tagging	

auto-advance-tagging (bool)[state: (bool)]

Description: Turn on/off the tagging auto advance feature. Toggles the state if Value is not provided. When auto advance is on once a tag is entered the tags panel advances to the next tab.

Example:

Delay	Shortcut	Value
	auto-advance-tagging	

toggle-tags-panel (bool)[state: (bool)]

Description: Will show or hide the tags panel based on Value. Toggles the visibility if Value is not provided.

Example:

Delay	Shortcut	Value
	toggle-tags-panel	

digit (int)

Description: Used to enter tags, specifically if "tagging" has been enabled this will perform the same function as typing in numbers on the Control Surface which will populate the dataview field. This should be combined with "tagging" and "enter" shortcuts. The tags are represented by the Value and follow the same two digit format you would use with the number pad on the control surface.

Example:

Delay	Shortcut	Value
	tagging	true
	digit	01
	enter	

select-prev-row

Description: Select the previous row in the Clip list or Play list based on the controlled output's current mode.

Example:

Delay	Shortcut	Value
	select-prev-row	

select-prev-row-shft

Description: Adds the previous row in the Clip list or Play list to the current selections based on the controlled output's current mode.

Example:

Delay	Shortcut	Value
	select-prev-row-shft	

select-prev-column

Description: Select the previous column in the Clip list or Play list based on the controlled output's current mode.

Example:

Delay	Shortcut	Value
	select-prev-column	

select-next-row

Description: Select the next row in the Clip list or Play list based on the controlled output's current mode.

Example:

Delay	Shortcut	Value
	select-next-row	

select-next-row-shft

Description: Adds the next row in the Clip list or Play list to the current selections based on the controlled output's current mode.

Example:

Delay	Shortcut	Value

	select-next-row-shft	

select-next-column

Description: Select the next column in the clip list or play list based on the controlled output's current mode.

Example:

Delay	Shortcut	Value
	select-next-column	

select-collection (int)

Description: Select a collection by index based on the current mode (Clips or Playlist).

Example:

Delay	Shortcut	Value
	select-collection	3

select-tags-collection (int)

Description: Select a tags collection by index. Index is in reference to Tag tabs such as the default TEAM, and PLAYER and the index numbering starts with 0.

Example:

Delay	Shortcut	Value
	select-tags-collection	1

cut

Description: Perform a cut operation on the currently selected items in the controlled output's current mode. For example, cuts the selected clips if the controlled output is in clip list mode.

Example:

Delay	Shortcut	Value
	cut	

copy

Description: Copy the currently selected items in the controlled output's current mode to the clipboard.

Example:

Delay	Shortcut	Value
	copy	

paste

Description: Paste items from the clipboard to the spreadsheet for the controlled output's current mode.

Example:

Delay	Shortcut	Value
	paste	

tbar (double)

Description: Set the tbar value for the controlled output which controls the playback speed. Valid values are between 0.0 and 1.0. Respects linked outputs (If the outputs are linked the playback speed is set for all outputs).

Example:

Delay	Shortcut	Value
	tbar	0.5

tbar-alt-mode (bool)

Description: Toggles the tbar alt mode on/off. When off the tbar range is mapped to playback speeds between 0 and 100%. When on the tbar range is mapped to playback speeds between -200% and +200%.

Example:

Delay	Shortcut	Value
	tbar-alt-mode	

publish

Description: Adds the currently selected items in the controlled output's current mode to the publish queue (e.g. for publishing to social media).

Example:

Delay	Shortcut	Value
	publish	

grab

Description: Captures a still image of all live camera inputs and a still image of what is currently on both outputs. The still capture of what is currently on the controlled output is added to the clip list.

Example:

Delay	Shortcut	Value
	grab	

grab_still

Description: Duplicate of "grab".

Example:

Delay	Shortcut	Value
	grab_still	

goto-clip

Description: Selects the clip with the ID that matches the current UTEB text on the controlled output.

Example:

Delay	Shortcut	Value
	goto-clip	

goto-angle (int)

Description: Switches the camera angle for whatever is on output. Value for Camera 1 is 1.

Example:

Delay	Shortcut	Value
	goto-angle	2

goto-angle-cue (int)

Description: The same as "goto-angle" but if the controlled output is in clip list mode then playback is restarted at the clip's In Point.

Example:

Delay	Shortcut	Value

	goto-angle-cue	

goto-angle-offset (int)

Description: Adjusts the camera angle for whatever is on output by Value delta. For example if you are on Camera 1 using a Value of 2 will change the angle to Camera 3. A value of -2 while on Camera 3 will change to Camera 1.

Example:

Delay	Shortcut	Value
	goto-angle-offset	1

search

Description: Searches for clips with comments that match the current UTEB text.

Example:

Delay	Shortcut	Value
	search	

show-angle-previews

Description: Toggle the visibility of the clip camera angle previews in the clip list.

Example:

Delay	Shortcut	Value
	show-angle-previews	

navigation (int)

Description: Navigation: left=0, up=1, right=2, down=3. Equivalent to cursor keys.

Example:

Delay	Shortcut	Value
	navigation	3

navigation-sel (int)

Description: Navigation selection: left=0, up=1, right=2, down=3. Equivalent to using the shift-cursor keys.

Example:

Delay	Shortcut	Value
	navigation-sel	1

D.2 Navigation

navigation-tabs (int)

Description: Traverse the tabs for the controlled output. Use a Value of 0 to select the previous tab and use a Value of 1 to select the next tab. If in “Clip mode” traverses the clip list tabs. If in “Playlist mode” traverses the playlist tabs.

Example:

Delay	Shortcut	Value
	navigation-tabs	1

navigation-tag-tabs (int)

Description: Traverse the tag tabs. Use a Value of 0 to select the previous tab and use a Value of 1 to select the next tab.

Example:

Delay	Shortcut	Value
	navigation-tag-tabs	1

goto-prev-clip

Description: Skip back transport control for both the clip list and playlist.

Example:

Delay	Shortcut	Value
	goto-prev-clip	

goto-next-clip

Description: Skip forward transport control for both the clip list and playlist.

Example:

Delay	Shortcut	Value
	goto-next-clip	

goto-in-point

Description: If the controlled output is stopped, seeks to the In point of the item currently on the controlled output.

Example:

Delay	Shortcut	Value
	goto-in-point	

goto-out-point

Description: If the controlled output is stopped, seeks to the Out point of the item currently on the controlled output.

Example:

Delay	Shortcut	Value
	goto-out-point	

dsk_auto (void)

Description: Fade the DSK on or off the controlled output.

Example:

Delay	Shortcut	Value
	dsk_auto	

dsk_take (void)

Description: Perform a "take" on the DSK for the controlled output on/off.

Example:

Delay	Shortcut	Value
	dsk_take	

_loop (bool)

Description: Set the playback looping for the targeted output On or Off. Use 0 to set Off and 1 to set On. If no Value is specified then looping is toggled.

Valid Prefixes:

- prev- Preview
- prog- Program

Example:

Delay	Shortcut	Value
	prog_loop	

D.3 Preview Row

prev_input_sel_num (string)[state: (string)]

Description: Selects a Playlist. Corresponds to the number keys on the 40CS that are delegated, for Value use corresponding input1, input2, input3, etc.

Example:

Delay	Shortcut	Value
	prev_input_sel_num	input3

prev_input_live_sel_num (string)[state: (string)]

Description: Select a live camera onto output A. Value represents the selected live camera, use "input1" for Camera 1, "input2" for Camera 2, etc.

Example:

Delay	Shortcut	Value
	prev_input_live_sel_num	input3

prev_input_clip_sel_num (string)[state: (string)]

Description: Select a Clip camera angle onto output A. Value represents the selected Clip camera, use "input1" for Camera 1, "input2" for Camera 2, etc.

Example:

Delay	Shortcut	Value
	prev_input_clip_sel_num	Input2

prev_input_list_sel_num (string)[state: (string)]

Description: Select a Playlist onto output A. Value represents the selected Playlist, use "input1" for Playlist 0, "input2" for Playlist 1, etc.

Example:

Delay	Shortcut	Value

	prev_input_list_sel_num	input2

D.4 Program Row

prog_input_sel_num (string)[state: (string)]

Description: Selects live input. Corresponds to the number keys on the 40CS that are delegated, for Value use corresponding input1, input2, input3, etc.

Example:

Delay	Shortcut	Value
	prog_input_sel_num	Input2

prog_input_live_sel_num (string)[state: (string)]

Description: Select a live camera onto output B. Value represents the live camera selected, use "input1" for Camera 1, "input2" for Camera 2, etc.

Example:

Delay	Shortcut	Value
	prog_input_live_sel_num	input2

prog_input_clip_sel_num (string)[state: (string)]

Description: Select a clip camera angle onto output B. Value represents the clip camera selected, use "input1" for Camera 1, "input2" for Camera 2, etc.

Example:

Delay	Shortcut	Value
	prog_input_clip_sel_num	input3

prog_input_list_sel_num (string)[state: (string)]

Description: Select a playlist onto output B. Value represents the playlist selected, use "input1" for Playlist 0, "input2" for Playlist 1, etc.

Example:

Delay	Shortcut	Value
	prog_input_list_sel_num	input2

Preview and Program Row Transport Controls	
Prefix	**Description**
prev	
prog	

_play (bool)

Description: Start playback on the targeted (prefix) output.

Example:

Delay	Shortcut	Value
	prog_play	

_stop (bool)

Description: Stop playback on the targeted (prefix) output.

Example:

Delay	Shortcut	Value
	prog_stop	

_skip_back (void)

Description: Skip back transport control for the playlist on the targeted (prefix) output.

Example:

Delay	Shortcut	Value
	prog_skip_back	

_skip_forward (void)

Description: Skip forward for the playlist on the targeted (prefix) output.

Example:

Delay	Shortcut	Value
	prog_skip_forward	

_set_play_speed (int)[state: (int)]

Description: Initiate playback at “Value” speed for the targeted (prefix) output. If the Value is equal to 25, then playback speed is at 25%. If no Value is specified, then playback speed defaults to 100%.

Example:

Delay	Shortcut	Value
	prog_set_play_speed	25

D.6 DSK Row Input Shortcuts

DSK Row Input Shortcuts	
Prefix	Details
dsk1, dsk2	

_sel_num (string)[state: (string)]

Description: Corresponds to the number keys on the 40CS that are delegated according to "dsk1_delegate" or "dsk2_delegate". Value represents selected live camera, use "input1" for Camera 1, "input2" for Camera 2, etc.

Example:

Delay	Shortcut	Value
	dsk1_sel_num	input1

_live_sel_num (string)[state: (string)]

Description: Set the targeted (prefix) DSK source to a live camera. Value represents selected live camera, use "input1" for Camera 1, "input2" for Camera 2, etc.

Example:

Delay	Shortcut	Value
	dsk2_live_sel_num	input2

_clip_sel_num (string)[state: (string)]

Description: Set the targeted (prefix) DSK source to a clip camera angle preview. Value represents the selected clip camera angle, use "input1" for Camera 1, "input2" for Camera 2, etc.

Example:

Delay	Shortcut	Value
	dsk_2_clip_sel_num	input3

_buffer_sel_num (string)[state: (string)]

Description: Set the targeted (prefix) DSK source to one of 8 frame buffers. Value represents the frame buffers, use “input1” for buffer 1, “input2” for buffer 2, etc.

Example:

Delay	Shortcut	Value
	dsk1_buffer_sel_num	input4

_sel_source (string)[state: (string)]

Description: Set the targeted (prefix) DSK source to "black", "net", or "output".

Example:

Delay	Shortcut	Value
	dsk2_sel_source	net

.

D.7 Transition Delegates

fx_auto (void)

Description: Delegates an auto transition to "main", "dsk1", "dsk2", or "ftb".

Example:

Delay	Shortcut	Value
	fx_auto	

fx_take (void)

Description: Delegates a take to "main", "dsk1", "dsk2", or "ftb".

Example:

Delay	Shortcut	Value
	fx_take	

fx_tbar_value (double)[state: (double)]

Description: Delegates tbar changes to "main", "dsk1", "dsk2", or "ftb".

Example:

Delay	Shortcut	Value
	fx_tbar_value	25

fx_adjust_fxrate (double)[state: (double)]

Description: Delegates FX rate adjustments to "main", "dsk1", "dsk2", or "ftb". Values entered will increment or decrement the current FX rate.

Example:

Delay	Shortcut	Value
	fx_adjust_fxrate	5

fx_next_fxrate_preset (void)

Description: Delegates cycling through FX rate presets to "main", "dsk1", "dsk2", or "ftb".

Example:

Delay	Shortcut	Value
	fx_next_fxrate_preset	

fx_sel_num (string)[state: (string)]

Description: Delegated to "dsk1_sel_num" or "dsk2_sel_num". Corresponds to numbered TriCaster 40 CS FX/Overlay row. Value represents the selected Clip List Preview, use "input1" for Camera 1, "input2" for Camera 2, etc.Will only work if a TriCaster 40CS is plugged in and Overlay Delegate DSK1 or DSK2 must be selected on the control surface.

Example:

Delay	Shortcut	Value
	fx_sel_num	input2

fx_live_sel_num (string)[state: (string)]

Description: Delegated to "dsk1_live_sel_num" or "dsk2_live_sel_num". Corresponds to numbered TriCaster 40 CS FX/Overlay row. Set the DSK source to a live camera. Value represents selected live camera, use "input1" for Camera 1, "input2" for Camera 2, etc Will only work if a TriCaster 40CS is plugged in and Overlay Delegate DSK1 or DSK2 must be selected on the control surface.

Example:

Delay	Shortcut	Value
	fx_live_sel_num	input3

fx_clip_sel_num (string)[state: (string)]

Description: Delegated to "dsk1_clip_sel_num" or "dsk2_clip_sel_num". Corresponds to numbered TriCaster 40 CS FX/Overlay row. Value represents the selected Clip List Preview, use "input1" for Camera 1, "input2" for Camera 2, etc.Will only work if a TriCaster 40CS is plugged in and Overlay Delegate DSK1 or DSK2 must be selected on the control surface.

Example:

Delay	Shortcut	Value
	fx_clip_sel_num	input2

fx_buffer_sel_num (string)[state: (string)]

Description: Delegated to "dsk1_buffer_sel_num" or "dsk2_buffer_sel_num". Corresponds to numbered TriCaster 40 CS FX/Overlay row. Set the DSK source to one of 8 frame buffers. Value represents the frame buffers, use “input1” for buffer 1, “input2” for buffer 2, etc. Will only work if a TriCaster 40CS is plugged in and and Overlay Delegate DSK1 or DSK2 must be selected on the control surface.

Example:

Delay	Shortcut	Value
	fx_buffer_sel_num	input8

fx_sel_source (string)[state: (string)]

Description: Delegated to "dsk1_sel_source" or "dsk2_sel_source". Set the DSK source to "black", "net", or "output". Will only work if a TriCaster 40CS is plugged in and and Overlay Delegate DSK1 or DSK2 must be selected on the control surface.

Example:

Delay	Shortcut	Value
	fx_sel_source	net

D.8 Transitions

40CS Transition Shortcuts	
Prefix	Details
dsk1, dsk2	
main	

_tbar_value (double)[state: (double)]

Description: Set the TBar location value.

Example:

Delay	Shortcut	Value
	main_tbar_value	50

_adjust_fxrate (double)[state: (double)]

Description: Adjust the FX rate, incrementing or decrementing based on Value.

Example:

Delay	Shortcut	Value
	main_adjust_fxrate	1

_next_fxrate_preset (void)

Description: Select the next Slow, Medium, or Fast preset.

Example:

Delay	Shortcut	Value
	main_next_fxrate_preset	

ftb_tbar_value (double)[state: (double)]

Description: Manual fade to/from black on the program output (Output B). Value range between 0.0 and 100.0.

Example:

Delay	Shortcut	Value
	ftb_tbar_value	75

ftb_auto (void)

Description: Fade black on/off the program output (Output B).

Example:

Delay	Shortcut	Value
	ftb_auto	

ftb_take (void)

Description: Immediately toggle black on/off program output (Output B).

Example:

Delay	Shortcut	Value
	ftb_take	

main_cycle_fxsel (int)[state: (int)]

Description: Iterate through the Main FX transition presets.

Example:

Delay	Shortcut	Value
	main_cycle_fxsel	1

main_toggle_fxreverse (void)

Description: Toggle reverse on/off for the Main FX transition.

Example:

Delay	Shortcut	Value
	main_toggle_fxreverse	

main_auto (void)

Description: This shortcut is used to trigger automatic transitioning between the Program and Preview rows.

Example:

Delay	Shortcut	Value
	main_auto	

main_take (void)

Description: Perform a Take from preview onto program.

Example:

Delay	Shortcut	Value
	main_take	

select-transition (int)

Description: Select a main transition preset by position (4800CS).

Example:

Delay	Shortcut	Value
	select-transition	3

mainbus (property_name string, property_value string)

Description: Allows you to select a transition from the from the palette in the Transition Control Area, set the duration of the transition and set “Ping Pong”, “Reverse”, and “AutoPlay” ON/OFF. The following are valid Property Names and their associated values.

- SelectedFXIndex-select a transition from the nine available in the pallet, valid values range from 0-8.
- FXDuration- set to SMF speeds using values 2, 1, 0.5.
- FXPingPongs- flag “Ping Pong” setting on using either True or False
- FXReversed- flag “Reverse” setting on using either True or False
- FXAutoPlays- flag “AutoPlay setting on using either True or False

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	mainbus		property_name	SelectedFXIndex	property_value	4

mainbus_set_transition_preset_at (index int, path string)

Description: Allows you to select a transition from the local drive and assign it to the palette in the Transition Control Area. Index refers to the nine preset slots and valid values range from 0-8. Path is the file path location on the local drive.

Example:

hortcut	Value	Key 0	Value 0	Key 1	Value 1
Mainbus_set_transition_preset_at		index	2	path	C:\3PLAY\Effects\Transitions\Blinds\Hard\Blinds 8 UL(H).trans

D.9 DSK

dsk1 (property_name string, property_value string)

Description: Allows you to set source and layer settings for the DSK.

Valid property_names and their property_values:

- Source- set the source for “DSK A”. Valid values are:

<table>
<tr><th>Valid Source “property_value”</th><th>UI Labeling</th></tr>
<tr><td>LiveCamera|0, LiveCamera|1…</td><td>Cameras</td></tr>
<tr><td>LowRezPlayback|0, LowRezPlayback|2…</td><td>Clip List Preview</td></tr>
<tr><td>FrameBuffer|0, Framebuffer|1…</td><td>Buffer</td></tr>
<tr><td>NetInputs</td><td>Network input</td></tr>
<tr><td>Output</td><td>Output B, Output A</td></tr>
<tr><td>Black</td><td>Black</td></tr>
</table>

- IsFadeOn- set “Use Fade transition” ON/OFF using True or False as the “property_value”.
- LengthMultiplier- set duration of Fade.
- EnablePositioning- set “Position” ON/OFF using True or False as the “property_value”.
- OffsetX - Used to translate the layer on the X axis supplied as Value.
- OffsetY - Used to translate the layer on the Y axis supplied as Value.
- ScaleX - Used to change the scale along the X axis supplied as Value.
- ScaleY – Used to change the scale along the Y axis supplied as Value.
- EnableCrop- set “Edges” ON/OFF using True or False as the “property_value”.
- CropLeft – Used to change the “Edges” left by Value.
- CropRight – Used to change the “Edges” right by Value.
- CropBottom – Used to change the “Edges” bottom by Value.
- CropTop – Used to change the “Edges” top by Value.

Example:

<table>
<tr><th>Delay</th><th>Shortcut</th><th>Value</th><th>Key 0</th><th>Value 0</th><th>Key 1</th><th>Value 1</th></tr>
<tr><td></td><td>dsk1</td><td></td><td>property_name</td><td>Source</td><td>property_value</td><td>LiveCamera|2</td></tr>
</table>

dsk2 (property_name string, property_value string)

Description: Allows you to set source and layer settings for the DSK.

Valid property_names and their property_values:

- Source- set the source for “DSK B”. Valid values are:

<table>
<tr><th>Valid Source “property_value”</th><th>UI Labeling</th></tr>
<tr><td>LiveCamera|0, LiveCamera|1…</td><td>Cameras</td></tr>
<tr><td>LowRezPlayback|0, LowRezPlayback|2…</td><td>Clip List Preview</td></tr>
</table>

FrameBuffer|0, Framebuffer|1…	Buffer
NetInputs	Network input
Output	Output B, Output A
Black	Black

- IsFadeOn- set “Use Fade transition” ON/OFF using True or False as the “property_value”.
- LengthMultiplier- set duration of Fade.
- EnablePositioning- set “Position” ON/OFF using True or False as the “property_value”.
- OffsetX - Used to translate the layer on the X axis supplied as Value.
- OffsetY - Used to translate the layer on the Y axis supplied as Value.
- ScaleX - Used to change the scale along the X axis supplied as Value.
- ScaleY – Used to change the scale along the Y axis supplied as Value.
- EnableCrop- set “Edges” ON/OFF using True or False as the “property_value”.
- CropLeft – Used to change the “Edges” left by Value.
- CropRight – Used to change the “Edges” right by Value.
- CropBottom – Used to change the “Edges” bottom by Value.
- CropTop – Used to change the “Edges” top by Value.

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	dsk2		property_name	EnablePositioning	property_value	True

DSKPrefix	Details
dsk1, dsk2	

_reset_position (void)

Description: Resets position values to origin.

Example:

Delay	Shortcut	Value
	dsk1_reset_position	

_reset_crop (void)

Description: Resets all crop values to origin. (Origin is 0.).

Example:

Delay	Shortcut	Value

	dsk2_reset_crop	

_auto (void)[state: (bool)]

Description: Will trigger an “auto”. The bool value is only used in the state, to light the “auto” button.

Example:

Delay	Shortcut	Value
	dsk1_auto	

_take(void)[state: (bool)]

Description: Will trigger a “take”. The bool value is only used in the state, to light the “take” button.

Example:

Delay	Shortcut	Value
	dsk2_take	

D.10 Network Input

net_input_sel(int)[state: (int)]

Description: Select an available network input connection using Value. Valid inputs will be referred to by their computer names on the network.

Example:

Delay	Shortcut	Value
	net_input_sel	XD450-TriCaster

D.11 AirSend API

Shortcuts Added for Airsend Command API	
Prefix	Details
out1, out2	

_clip_select (string)

Description: Select clip if Value is a clip ID or a playlist if Value is a playlist index.

Example:

Delay	Shortcut	Value
	out1_clip_select	2-2

_clip_move (int)

Description: Skip back/forward transport control for both the clip list and playlist using Value for the amount moved.

Example:

Delay	Shortcut	Value
	out2_clip_move	2

_position_play (position double, speed double)

Description: Will position and/or play the targeted (prefix) output. If no speed is provided the current playback speed is maintained.

- **Position**- Optional offset from the In point. Ignored if the prefix output is in Live or Delayed mode.
- **Speed**- Speed multiplier between -2.0 and 2.0.

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	out1_position_play		position	8	speed	1.5

_jog (double)

Description: Set the Jog distance for the targeted (prefix) output using Value.

Example:

Delay	Shortcut	Value
	out2_jog	5

D.12 AMP

Shortcuts Added for AMP	
Prefix	Details
amp_vtr1, amp_vtr2	

_loop (bool)

Description: Changes playback looping for the prefix AMP playout channel. If no shortcut value is specified then looping is toggled.

Example:

Delay	Shortcut	Value
	amp_vtr1_loop	

_play(void)

Description: Plays at 1X speed the prefix AMP playout channel.

Example:

Delay	Shortcut	Value
	amp_vtr2_play	

_stop (void)

Description: Stops playback of the prefix AMP playout channel.

Example:

Delay	Shortcut	Value
	amp_vtr2_stop	

_cue (string)

Description: Cues the results of a previous query on output of the prefixed channel.

Example:

Delay	Shortcut	Value
	amp_vtr1_cue	

_query (string)

Description: Sends a unique string describing what is currently on output of the prefix AMP playout channel to the FC3 server with name Value.

Example:

Delay	Shortcut	Value
	amp_vtr1_query	

_eject (void)

Description: Clears the prefix AMP VTR contents.

Example:

Delay	Shortcut	Value
	amp_vtr2_eject	

D.13 Publish

share_add(filename string, title string, comment string, mediasite string, siteid string, inpoint double, outpoint double, asstill bool, upload bool, presets string)

Description: Publish.

- **Filename**- Required
- **Title**- Title
- **Comment**- Comment
- **Mediasite**- If no site entered, use last used
- **Siteid**- this is a string representation of the GUID of the plugin id
- **Inpoint**- In point in seconds
- **Outpoint**- Out point in seconds
- **Asstill**- Set to true to send a still grab of the clip instead. Default to false if empty
- **Upload**- Set to true to upload immediately. Default to false if empty
- **Presets**- Preset names can be used as default selected values. Separate the preset names using a pipe.

Example:

Shortcut	Value	Key 0	Value 0	Key 1	Value 1	Key 2	Value 2
share_add		Filename	D:\media\londonbus.mov	title	London Bus	comment	London Bus

Key 3	Value 3	Key 4	Value 4	Key 5	Value 5	Key 6	Value 6	Key 7	Value 7
inpoint	0	outpoint	10.01	asstill	false	upload	false	presets	My YouTube| My Facebook

D.14 Transfer

add_to_clips_tab (tab_index int, item_id string, type string)

Description: Copy clips from one clip list to another or copy clip camera angles to a playlist.

- Tab_index- The destination tab index.
- Item_id- The event ID(s) of the clip(s) to copy. Must include the camera angle for recorded clips being copied to a playlist. Multiple items can be combined with pipes.
- Type- Use "cliplist" if the destination is another clip list. Use "playlist" if the copy destination is a playlist.

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1	Key 2	Value 2
	add_to_clips_tab		tab_index	6	item_id	2-0|2-2	type	cliplist

add_to_playlist_tab (tab_index int, item_id string)

Description: Copy playlist items from one playlist to another.

- Tab_index- The destination of the items being copied. Numeration starts at 0.
- Item_id- Describes the playlist items being copied. Reference items by their "tab index and item index" format as follows:

  "tab index"-"item index"

  In both cases the numeration starts at 0. Multiple items can be listed and separated by pipes.

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	add_to_playlist_tab		tab_index	7	item_id	1-0|1-5

D.15 Tabs

add_new_tab (type string, tab_name string)

Description: Add a new Playlist or Tagslist tab and name it.

- Type- Use "playlist" as a value to add a new playlist tab. Use "tagslist" as a Value to add a new Tags Tab

- Tab_name- Provide a name for the new tab.

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	Add_new_tab		type	playlist	tab_name	A new Tab

delete_tab (type string, tab_index int)

Description: Delete a Playlist or Tagslist tab.

- Type- "playlist" to delete a playlist tab. "tagslist" to delete a Tags Tab

- Tab_index- The index of the tab to be deleted, numeration starts at 0.

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	delete_tab		type	tagslist	tab_index	4

D.16 Miscellaneous

find

Description: Show the search results across all clip lists for the text currently entered into the UTEB.

Example:

Delay	Shortcut	Value
	find	

import-media

Description: Opens the media browser.

Example:

Delay	Shortcut	Value
	import-media	

goto-timecode

Description: Will jump to the timecode currently entered into the UTEB and put the 3Play in Delayed playback mode.

Example:

Delay	Shortcut	Value
	goto-timecode	

import_file_clipslist (string)

Description: Imports file(s) found at the path(s) specified by Value into the current clip list for the controlled output. Import multiple files by separating the paths with pipes.

Example:

Delay	Shortcut	Value
	import_file_clipslist	D:\Media\Clips\Fountain.mov

import_file_playlist (string)

Description: Imports file(s) found at the path(s) specified by Value into the current playlist for the controlled output. Import multiple files by separating the paths with pipes.

Example:

Delay	Shortcut	Value
	import_file_playlist	D:\Media\Clips\Fountain.mov

import_file_music (string)

Description: Imports the music at the path specified in Value. It will be imported into the “Music Track” list for the current playlist of the controlled output.

Example:

Delay	Shortcut	Value
	import_file_music	D:\Media\Soundand Music\myAudioFile.mp3

goto_clip_tab(int)

Selects the clip tab specified by {value}. The {value} corresponds to the tab index.

goto_playlist_tab(int)

Selects the playlist tab specified by {value}. The {value} corresponds to the tab index.

switcher_finish_and_send_changes (void)

Description: “switcher_begin_changes” intercepts all messages to the switcher until “switcher_finish_and_send_changes” is sent. Allows 'batching' of switcher commands.

Example:

Delay	Shortcut	Value
	switcher_finish_and_send_changes	

display_version

Description: Display the software build version information in the titlebar.

Example:

Delay	Shortcut	Value
	display_version	

set_router_mapping (router_input int, system_input int)

Description: Map router input to a TriCaster or 3Pplay input. By default mapping is normally set 1:1 but you can be more specific. For example if you need to set a different range starting at 10 instead of 1, the router_input would be 10 and the system_input would be 1.

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	set_router_mapping		router_input	1	system_input	1

set_item_prop (type string, item_id string, prop string, value string)

Description: Modify properties of clips or playlist items.

- **Type**- "cliplist" to change a clip's property. "playlist" to change a property of a playlist item.
- **Item_id**- If not provided then sets properties on the selection according to the type. If type is "cliplist", Value is the Event ID. If type is "playlist", Value is "tab index"-"item index".
- **Prop**- The name of the property to change. Valid clip values are "InTC", "OutTC", "Duration", "AudioLevel", "Alias[0]", "Alias[1]", etc. Valid playlist item values are "Name", "FrontAudioEnabled", "RearAudioEnabled", "FrontAudioVolume", "RearAudioVolume", "InTimecode", "Duration", "Speed", "TransitionIndex", "TransitionMultiplier", and "TransitionDuration".
- **Value**- A value interpreted based on the specified "prop".

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1	Key 2	Value 2	Key 3	Value 3
	set_item_prop		type	playlist	item_id	1-4	prop	Speed	value	50

delete_item (type string, item_id string)

Description: Delete clips or playlist items.

- **Type**- Use "clipslist" to delete a clip or "playlist" to delete a playlist item.
- **Item_id**- If an id is not provided then the selection will be deleted according to the type. If type is "clipslist", Value is the Event ID. If type is "playlist", Value is "tab index"-"item index".

Example:

Delay	Shortcut	Value	Key 0	Value 0	Key 1	Value 1
	delete_item		type	clipslist	item_id	0-2

APPENDIX E. DATALINK HARDWARE KEYS

This section lists the actual key names that are available for use with DataLink for the different brands of external equipment it supports. Mostly, the key names are self-explanatory, but we've added slightly more descriptive notes where appropriate. The list is grouped by manufacturer.

Note: the key names listed are shown inserted between percent (%) signs as a reminder, since this is how you will enter them onto your pages.

E.1 DAKTRONICS

E.1.1 BASEBALL

%DakClock%	-	Game Clock Time – "MM:SS.T"
%DakClockStatus%	-	Game Clock Status
%DakHomeHits%	-	Home Team Hits
%DakGuestScore%	-	Guest Team Score
%DakInning%	-	Current inning
%DakHhr%	-	Hour (from Clock Time)
%DakMin%	-	Minutes (from Clock Time)
%DakSec%	-	Seconds (from Clock Time)
%DakTen%	-	Tenths (secs/10 from Clock Time)

E.1.2 BASKETBALL

%DakClock%	-	Game Clock Time – "MM:SS.T"
%DakClockStatus%	-	Game Clock Status
%DakShotClock%	-	Shot Clock Time – "SS"
%DakHomeScore%	-	Home Team Score
%DakGuestScore%	-	Guest Team Score
%DakHomeFouls%	-	Home Team Fouls
%DakGuestFouls%	-	Guest Team Fouls
%DakHomeTOFull%	-	Home Time Outs Left – Full
%DakHomeTOPart%	-	Home Time Outs Left – Partial
%DakHomeTOTotal%	-	Home Time Outs Left – Total
%DakGuestTOFull%	-	Guest Time Outs Left – Full
%DakGuestTOPart%	-	Guest Time Outs Left – Partial
%DakGuestTOTotal%	-	Guest Time Outs Left – Total
%DakPeriod%	-	Current period
%DakHhr%	-	Hour (from Clock Time)
%DakMin%	-	Minutes (from Clock Time)
%DakSec%	-	Seconds (from Clock Time)
%DakTen%	-	Tenths (secs/10 from Clock Time)

E.1.3 FOOTBALL

%DakClock%	-	Game Clock Time – “MM:SS.T”
%DakClockStatus%	-	Game Clock Status
%DakPlayClock%%	-	Play Clock Time – “SS“
%DakHomeScore%	-	Home Team Score
%DakGuestScore%	-	Guest Team Score
%DakHomeTOFull%	-	Home Time Outs Left – Full
%DakHomeTOPart%	-	Home Time Outs Left – Partial
%DakHomeTOTotal%	-	Home Time Outs Left – Total
%DakGuestTOFull%	-	Guest Time Outs Left – Full
%DakGuestTOPart%	-	Guest Time Outs Left – Partial
%DakGuestTOTotal%	-	Guest Time Outs Left – Total
%DakQuarter%	-	Current quarter
%DakMin%	-	Minutes (from Clock Time)
%DakSec%	-	Seconds (from Clock Time)
%DakTen%	-	Tenths (secs/10 from Clock Time)

E.1.4 HOCKEY

%DakClock%	-	Game Clock Time – “MM:SS.T”
%DakClockStatus%	-	Game Clock Status
%DakShotClock%%	-	Shot Clock Time – “SS“
%DakHomeScore%	-	Home Team Score
%DakGuestScore%	-	Guest Team Score
%DakHomeTOFull%	-	Home Time Outs Left – Full
%DakHomeTOTotal%	-	Home Time Outs Left – Total
%DakGuestTOFull%	-	Guest Time Outs Left – Full
%DakGuestTOTotal%	-	Guest Time Outs Left – Total
%DakPeriod%	-	Current period
%DakMin%	-	Minutes (from Clock Time)
%DakSec%	-	Seconds (from Clock Time)
%DakTen%	-	Tenths (secs/10 from Clock Time)

E.1.5 SOCCER

%DakClock%	-	Game Clock Time – “MM:SS.T”
%DakClockStatus%	-	Game Clock Status
%DakShotClock%%	-	Shot Clock Time – “SS“
%DakHomeScore%	-	Home Team Score
%DakGuestScore%	-	Guest Team Score
%DakHomeTOFull%	-	Home Time Outs Left – Full
%DakGuestTOFull%	-	Guest Time Outs Left – Full
%DakGuestTOTotal%	-	Guest Time Outs Left – Total

%DakHalf%	-	Current half
%DakMin%	-	Minutes (from Clock Time)
%DakSec%	-	Seconds (from Clock Time)
%DakTen%	-	Tenths (secs/10 from Clock Time)

E.1.6 VOLLEYBALL

%DakClock%	-	Game Clock Time – “MM:SS.T”
%DakClockStatus%	-	Game Clock Status
%DakHomeServiceIndicator%		
%DakHomeScore%	-	Home Team Score
%DakGuestScore%	-	Guest Team Score
%DakHomeTOFull%	-	Home Time Outs Left – Full
%DakHomeTOTotal%	-	Home Time Outs Left – Total
%DakGuestTOFull%	-	Guest Time Outs Left – Full
%DakGuestTOTotal%	-	Guest Time Outs Left – Total
%DakGameNumber%	-	Current game number
%DakMin%	-	Minutes (from Clock Time)
%DakSec%	-	Seconds (from Clock Time)
%DakTen%	-	Tenths (secs/10 from Clock Time)

E.2 DAKTRONICS CG

E.2.1 BASEBALL

%CGDakHomeScore%	-	Home Team Score
%CGDakGuestScore%	-	Guest Team Score
%CGDakInning%	-	Current inning
%CGDakInningText%	-	Current inning (text)
%CGDakInningDescription%	-	Inning Description (text)
%CGDakHomeAtBat%	-	Home At-bat indicator (0 or 1).
%CGDakGuestAtBat%	-	Guest At-bat indicator (0 or 1).
%CGDakHomeHits%	-	Home Team Hits
%CGDakHomeErrors%	-	Home Team Errors
%CGDakHomeLeftOnBase%	-	Home Team Left-on-base
%CGDakGuestHits%	-	Guest Team Hits
%CGDakGuestErrors%	-	Guest Team Errors
%CGDakGuestLeftOnBase%	-	Guest Team Left-on-base
%CGDakBatterNumber%	-	At-bat Player Number
%CGDakBatterAverage%	-	At-bat Player Average
%CGDakBall%	-	Ball count
%CGDakStrike%	-	Strike count
%CGDakOut%	-	Outs
%CGDakHit%	-	Hits

%CGDakError%	-	Errors
%CGDakHitErrorText%	-	Error (text)
%CGDakErrorPosition%	-	Error Position
%CGDakInningLabel1%	-	First Inning label
%CGDakInningLabel2%	-	etc.
%CGDakInningLabel3%		
%CGDakInningLabel4%		
%CGDakInningLabel5%		
%CGDakInningLabel6%		
%CGDakInningLabel7%		
%CGDakInningLabel8%		
%CGDakInningLabel9%		
%CGDakInningLabel10%		
%CGDakInningLabel11%		
%CGDakInningLabel12%		
%CGDakHomeInningScore1%	-	Home Score, First Inning
%CGDakHomeInningScore2%	-	etc.
%CGDakHomeInningScore3%		
%CGDakHomeInningScore4%		
%CGDakHomeInningScore5%		
%CGDakHomeInningScore6%		
%CGDakHomeInningScore7%		
%CGDakHomeInningScore8%		
%CGDakHomeInningScore9%		
%CGDakHomeInningScore10%		
%CGDakHomeInningScore11%		
%CGDakHomeInningScore12%		
%CGDakGuestInningScore1%	-	Guest Score, First Inning
%CGDakGuestInningScore2%	-	etc.
%CGDakGuestInningScore3%		
%CGDakGuestInningScore4%		
%CGDakGuestInningScore5%		
%CGDakGuestInningScore6%		
%CGDakGuestInningScore7%		
%CGDakGuestInningScore8%		
%CGDakGuestInningScore9%		
%CGDakGuestInningScore10%		
%CGDakGuestInningScore11%		
%CGDakGuestInningScore12%		
%CGDakHomePitcherNum%	-	Home Pitcher Player Number
%CGDakHomePitchesBalls%	-	Home Pitches, Balls
%CGDakHomePitchesStrikes%	-	Home Pitches, Strikes
%CGDakHomePitchesFoulBall%	-	Home Pitches, Foul Balls

%CGDakHomePitchesInPlay%	-	Home Pitches In Play
%CGDakHomePitchesTotal%	-	Total Home Pitches
%CGDakGuestPitcherNum%	-	Guest Pitcher Player Number
%CGDakGuestPitchesBalls%	-	Guest Pitches, Balls
%CGDakGuestPitchesStrikes%	-	Guest Pitches, Strikes
%CGDakGuestPitchesFoulBall%	-	Guest Pitches, Foul Balls
%CGDakGuestPitchesInPlay%	-	Guest Pitches In Play
%CGDakGuestPitchesTotal%	-	Total Guest Pitches

E.2.2 BASKETBALL

%CGDakClock%	-	Game Clock Time – "MM:SS.T"
%CGDakClockStatus%	-	Game Clock Status
%CGDakShotClock%	-	Shot Clock Time – "SS“
%CGDakHomeScore%	-	Home Team Score
%CGDakGuestScore%	-	Guest Team Score
%CGDakHomeFouls%	-	Home Team Fouls
%CGDakGuestFouls%	-	Guest Team Fouls
%CGDakHomeTOFull%	-	Home Time Outs Left – Full
%CGDakHomeTOPart%%	-	Home Time Outs Left – Partial
%CGDakHomeTOTotal%	-	Home Time Outs Total
%CGDakGuestTOFull%	-	Guest Time Outs Left – Full
%CGDakGuestTOPart%	-	Guest Time Outs Left – Partial
%CGDakGuestTOTotal%	-	Guest Time Outs Left – Total
%CGDakPeriod%	-	Current period
%CGDakMin%	-	Minutes (from Clock Time)
%CGDakSec%	-	Seconds (from Clock Time)
%CGDakTen%	-	Tenths (secs/10 from Clock Time)

E.2.3 FOOTBALL

%CGDakClock%	-	Game Clock Time – "MM:SS.T"
%CGDakHomeTeamName%	-	Home Team Name
%CGDakGuestTeamName%	-	Guest Team Name
%CGDakHomeScore%	-	Home Team Score
%CGDakGuestScore%	-	Guest Team Name
%CGDakQuarter%	-	Current quarter
%CGDakBallOn%	-	Current ball position
%CGDakDown%	-	Current down
%CGDakToGo%	-	Yards to go
%CGDakHomePossess%	-	Possession indicator (0 or 1).
%CGDakGuestPossess%	-	Possession indicator (0 or 1).
%CGDakPlayClock%	-	Play Clock Time – "SS“
%CGDakHomeTO%	-	Home Time Outs

%CGDakGuestTO%	-	Guest Time Outs
%CGDakMin%	-	Minutes (from Clock Time)
%CGDakSec%	-	Seconds (from Clock Time)
%CGDakTen%	-	Tenths (secs/10 from Clock Time)

E.2.4 HOCKEY

%CGDakClock%	-	Game Clock Time – “MM:SS.T”
%CGDakClockStatus%	-	Game Clock running status indicator
%CGDakHomeScore%	-	Home Team Score
%CGDakGuestScore%	-	Guest Team Score
%CGDakHomeTO%	-	Home Time Outs
%CGDakGuestTO%%	-	Guest Time Outs
%CGDakHomeShotsOnGoal%	-	Home Shots on Goal
%CGDakGuestShotsOnGoal%	-	Guest Shots on Goal
%CGDakPeriod%	-	Current period
%CGDakHomePenalty1_PlayerNum%	-	Home Penalty, player number
%CGDakHomePenalty1_PenaltyTime%	-	Home Penalty, time left
%CGDakGuestPenalty1_PlayerNum%	-	Guest Penalty, player number
%CGDakGuestPenalty1_PenaltyTime%	-	Guest Penalty, time left
%CGDakHomePenalty2_PlayerNum%	-	Home Penalty, player number
%CGDakHomePenalty2_PenaltyTime%	-	Home Penalty, time left
%CGDakGuestPenalty2_PlayerNum%	-	Guest Penalty, player number
%CGDakGuestPenalty2_PenaltyTime%	-	Guest Penalty, time left
%CGDakMin%	-	Minutes (from Clock Time)
%CGDakSec%	-	Seconds (from Clock Time)
%CGDakTen%	-	Tenths (secs/10 from Clock Time)

E.2.5 SOCCER

%CGDakClock%	-	Game Clock Time – “HH:MM:SS.T”
%CGDakHomeTeamName%	-	Home Team Name
%CGDakGuestTeamName%	-	Guest Team Name
%CGDakHomeScore%	-	Home Team Score
%CGDakGuestScore%	-	Guest Team Score
%CGDakHalf%		-Current half
%CGDakHomeShotsOnGoal%	-	Home Shots on Goal
%CGDakHomeSaves%	-	Home Saves
%CGDakHomeCornerKicks%	-	Home Corner Kicks
%CGDakGuestShotsOnGoal%	-	Guest Shots on Goal
%CGDakGuestSaves%	-	Guest Saves
%CGDakGuestCornerKicks%	-	Guest Corner Kicks
%CGDakHomeFouls%	-	Home Fouls
%CGDakGuestFouls%	-	Guest Fouls

%CGDakHhr%	-	Hours (from Clock Time)
%CGDakMin%	-	Minutes (from Clock Time)
%CGDakSec%	-	Seconds (from Clock Time)
%CGDakTen%	-	Tenths (secs/10 from Clock Time)

E.2.6 VOLLEYBALL

%CGDakClock%	-	Game Clock Time – "MM:SS.T"
%CGDakClockStatus%		Game clock running status indicator
%CGDakHomeGameScore%	-	Home Team Score
%CGDakGuestGameScore%	-	Guest Team Score
%CGDakHomeTO%	-	Home Time Out
%CGDakGuestTO%	-	Guest Time Out
%CGDakHomeServiceIndicator%	-	Home Service indicator (0 or 1)
%CGDakGuestServiceIndicator%	-	Guest Service indicator (0 or 1)
%CGDakHomeGamesWon%	-	Home Games Won
%CGDakGuestGamesWon%	-	Guest Games Won
%CGDakGameNumber%		-Current Game Number
%CGDakHomeGameScore1%	-	Home Score, First Game
%CGDakHomeGameScore2%	-	Home Score, Second Game
%CGDakHomeGameScore3%	-	Home Score, Third Game
%CGDakHomeGameScore4%	-	Home Score, Fourth Game
%CGDakGuestGameScore1%	-	Guest Score, First Game
%CGDakGuestGameScore2%	-	Guest Score, Second Game
%CGDakGuestGameScore3%	-	Guest Score, Third Game
%CGDakGuestGameScore4%	-	Guest Score, Fourth Game
%CGDakMin%	-	Minutes (from Clock Time)
%CGDakSec%	-	Seconds (from Clock Time)
%CGDakTen%	-	Tenths (from Clock Time)

E.3 DSI KEYS:

E.3.1 BASKETBALL

%DSIClock%	-	Game Clock Time – "MM:SS.T"
%DSIShotClock%	-	Shot Clock Time – "SS"
%DSIMin%	-	Minutes (from Clock Time)
%DSISec%	-	Seconds (from Clock Time)
%DSITen%	-	Tenths (secs/10 from Clock Time)

E.4 OES

E.4.1 BASKETBALL

%OESClock%	-	Game Clock Time – “MM:SS.T”
%OESShotClock%	-	Shot Clock Time
%OESAwayScore%	-	Guest Team Score
%OESHomeScore%	-	Home Team Score
%OESHomeFouls%	-	Home Team Fouls
%OESAwayFouls%	-	Guest Team Fouls
%OESHomeTOFull%	-	Home Team Time Out - Full
%OESHomeTOPart%	-	Home Team Time Out - Partial
%OESAwayTOFull%	-	Guest Team Time Out - Full
%OESAwayTOPart%	-	Guest Team Time Out - Partial
%OESPeriod%	-	Current period
%OESMin%	-	Minutes (from Clock Time)
%OESSec%	-	Seconds (from Clock Time)
%OESTen%	-	Tenths (secs/10 from Clock Time)

E.5 Translux Fairplay

E.5.1 FOOTBALL

%TLFPClock%	-	Game Clock Time – “MM:SS.T”
%TLFPQuarter%	-	Current quarter
%TLFPHomeScore%	-	Home Team Score
%TLFPVisitorScore%	-	Visiting Team Score
%TLFPDown%	-	Current down
%TLFPToGo%	-	To go (yards
%TLFPBallOn%	-	Ball on (yard line)
%TLFPFieldTimer%	-	Current field timer (SS)
%TLFPMin%	-	Minutes (from Clock Time)
%TLFPSec%	-	Seconds (from Clock Time)
%TLFPTen%	-	Tenths (secs/10 from Clock Time)

E.6 Whiteway

E.6.1 BASKETBALL

%WWPeriod%	-	Current period
%WWClock%	-	Game Clock Time – “MM:SS.T”
%WWAwayScore%	-	Guest Team Score
%WWHomeScore%	-	Home Team Score
%WWShotClock%	-	Shot Clock Time

%WWMin%	-	Minutes (from Clock Time)
%WWSec%	-	Seconds (from Clock Time)
%WWTen%	-	Tenths (secs/10 from Clock Time)

E.7 Whiteway Rainbow

E.7.1 BASKETBALL

%WWRSportNum%	-	Sport Number
%WWRPeriod%	-	Current period
%WWRShotClock%	-	Shot Clock Time
%WWRAwayScore%	-	Guest Team Score
%WWRHomeScore%	-	Home Team Score
%WWRMinutes%	-	Minutes (from Clock Time)
%WWRSeconds%	-	Seconds (from Clock Time)
%WWRTenths%	-	Tenths (from Clock Time)

INDEX

Chapter 10 CREDITS

Acknowledgments:

Tim Jenison, Jim Plant

Engineering:

Andrew Cross, Alvaro Suarez, Brian Brice, Cary Tetrick, Charles Steinkuehler, Dan Fletcher, Gil Triana, Greg Heine, Jagannadh Malla, James Killian, Jan Uribe, Jarrod Davis, Jeremy Brosius, Jeremy Wiseman, John Perkins, Karen Zipper, Kevin Rouviere, Kirk Morger, Liviu Corsatea, Mahdi Mohajer, Masaaki Konno, Menghua Wang, Michael Joiner, Michael Watkins, Mike Murphy, Nathan Kovner, Naveen Jayakumar, Ryan Hansberger, Shawn Wisniewski, Steve Bowie, Todd Bryant, Troy Stevenson

Additional thanks to:

NewTek Marketing, Sales, Business Development, Customer Support, Training and Development, and Operations departments and personnel.